

307657

DEPARTMENT OF MECHANICAL ENGINEERING AND MECHANICS  
COLLEGE OF ENGINEERING AND TECHNOLOGY  
OLD DOMINION UNIVERSITY  
NORFOLK, VIRGINIA 23529

**GRID GENERATION AND FLOW COMPUTATION  
ABOUT A MARTIAN ENTRY VEHICLE**

By

J. E. Stewart, Graduate Research Assistant

and

S. N. Tiwari, Principal Investigator

Progress Report

For the period ended June 30, 1990

Prepared for  
National Aeronautics and Space Administration  
Langley Research Center  
Hampton, Virginia 23665

Under

Research Grant NCC1-68

Dr. Robert E. Smith, Jr., Technical Monitor  
ACD-Computer Applications Branch

(NASA-CR-107032) GRID GENERATION AND FLOW  
COMPUTATION ABOUT A MARTIAN ENTRY VEHICLE  
Progress Report, period ending 30 Jun. 1990  
(Old Dominion Univ.) 86 p

CSCL 228

N91-11042

Unclass

65/18 0302067

October 1990

DEPARTMENT OF MECHANICAL ENGINEERING AND MECHANICS  
COLLEGE OF ENGINEERING AND TECHNOLOGY  
OLD DOMINION UNIVERSITY  
NORFOLK, VIRGINIA 23529

**GRID GENERATION AND FLOW COMPUTATION  
ABOUT A MARTIAN ENTRY VEHICLE**

By

J. E. Stewart, Graduate Research Assistant

and

S. N. Tiwari, Principal Investigator

Progress Report  
For the period ended June 30, 1990

Prepared for  
National Aeronautics and Space Administration  
Langley Research Center  
Hampton, Virginia 23665

Under  
Research Grant NCC1-68  
Dr. Robert E. Smith, Jr., Technical Monitor  
ACD-Computer Applications Branch

Submitted by the  
Old Dominion University Research Foundation  
P.O. Box 6369  
Norfolk, Virginia 23508-0369

October 1990

## ABSTRACT

# GRID GENERATION AND FLOW COMPUTATION ABOUT A MARTIAN ENTRY VEHICLE

A number of vehicles are currently being proposed for a manned mission to Mars. One of these vehicles has a modified blunt-nosed cone configuration. Experimental results have been obtained for this vehicle in 1968. These results show lift-over-drag ratios comparable to those needed for Mars entry. Computations are performed here to verify the earlier results and to further describe the flight characteristics of this vehicle.

An analytical method is used to define the surface of this vehicle. A single-block volume grid is generated around the vehicle using an algebraic Two-Boundary Grid Generation algorithm (TBGG) and transfinite interpolation. Euler solutions are then obtained from a Langley Aerothermodynamic Upwind Relaxation Algorithm (LAURA) at Mach 6.0 and angles of attack of 0, 6, and 12 degrees.

The lift coefficient determined from the LAURA code agrees very well with the experimental results obtained in the 1968 study of this vehicle. The drag and pitching moment coefficients, however, are underestimated by the code since viscous effects are not considered. Contour plots of the flowfield show no evidence of separation for angles of attack up to 12 degrees.

### **ACKNOWLEDGMENTS**

This is a progress report on the research project "Numerical Solutions of Three-Dimensional Navier-Stokes Equations for Closed-Bluff Bodies" for the period ended June 30, 1990. Specific efforts during this period were directed in the area of "Grid Generation and Flow Computation About a Martian Entry Vehicle."

This work was supported by the NASA Langley Research Center through Cooperative Agreement NCC1-68. The cooperative agreement was monitored by Dr. Robert E. Smith, Jr., of the Analysis and Computation Division (Computer Applications Branch), NASA Langley Research Center, Mail Stop 125.

# TABLE OF CONTENTS

	<u>page</u>
ACKNOWLEDGEMENTS .....	ii
TABLE OF CONTENTS .....	iii
LIST OF FIGURES .....	v
LIST OF SYMBOLS .....	vii
Chapter	
1. INTRODUCTION .....	1
2. GRID GENERATION .....	5
2.1 Introduction .....	5
2.2 Surface Definition .....	5
2.3 Nose Discontinuity .....	14
2.4 Volume Grid .....	24
3. GOVERNING EQUATIONS AND SOLUTION TECHNIQUE .....	35
3.1 Introduction .....	35
3.2 Governing Equations .....	35
3.3 Boundary Conditions .....	41
3.4 Aerodynamic Quantities .....	43
4. FLOWFIELD COMPUTATIONS AND DISCUSSION OF RESULTS .....	45
4.1 Introduction .....	45
4.2 Cases Studied .....	46
4.3 Comparisons .....	51

5. CONCLUSIONS .....	58
REFERENCES .....	59
APPENDIX A: SURFACE GENERATION PROGRAMS: ZFIND, SHIP .....	61

# LIST OF FIGURES

<u>Figure</u>	<u>page</u>
2.1 Original model details (all dimensions in cm.)	6
2.2 Original equipment arrangement	7
2.3 Nose, forebody, and afterbody locations	8
2.4 Cross-sections of the vehicle: (a)circle, (b)egg-shaped ellipse, and (c)ellipse	10
2.5 Forebody or afterbody cross-section	11
2.6 Side and top view of surface grid	15
2.7 Initial nose configuration	16
2.8 Filling in discontinuity	18
2.9 Final nose configuration	20
2.10 Forebody/afterbody junction	21
2.11 Surface discontinuity: (a)before smoothing, and (b)after smoothing	22
2.12 New model details	23
2.13 Top, bottom, side, and exit grids	27
2.14 Definition of cap: (a)x-y plane, and (b)y-z plane	29
2.15 Definition of cap in the physical space	30
2.16 Intermediate surfaces resulting from transfinite interpolation	33
2.17 Mapping from physical domain to computational domain	34
3.1 Computational cell	36
3.2 Wall boundary	42
4.1 Hypersonic flow around a blunt body	47
4.2 $C_p$ contours for Mach 6.0, $\alpha = 0^\circ$	48

4.3 $C_p$ contours for Mach 6.0, $\alpha = 6^\circ$ .....	50
4.4 $C_p$ contours for Mach 6.0, $\alpha = 12^\circ$ .....	52
4.5 Coefficient of lift vs. angle of attack, Mach 6.0 .....	54
4.6 Coefficient of drag vs. angle of attack, Mach 6.0 .....	54
4.7 Coefficient of pitching-moment vs. angle of attack, Mach 6.0 .....	55
4.8 Lift-over-drag vs. angle of attack, Mach 6.0 .....	55
4.9 Density vs. distance along stagnation line, Mach 6.0, $\alpha = 0^\circ$ .....	56
4.10 $C_p$ vs. distance along z-axis, top of vehicle, Mach 6.0 .....	56
4.11 $C_p$ vs. distance along z-axis, bottom of vehicle, Mach 6.0 .....	57
4.12 $C_p$ vs. distance along z-axis, side of vehicle, Mach 6.0, $\alpha = 0^\circ$ .....	57

# LIST OF SYMBOLS

$A$	base area of vehicle
$A, B, C$	inviscid flux Jacobians
$a$	ellipse minor axis
$a_w$	weighting parameter
$a_1, a_2, a_3$	quadratic coefficients
$\vec{a}_i, \vec{b}_i, \vec{c}_i$	transfinite interpolation surfaces
$b$	ellipse major axis
$b_w$	weighting parameter
$C$	speed of sound
$C^i$	$i^{th}$ derivative continuity
$C_d$	drag coefficient
$C_l$	lift coefficient
$C_m$	pitching-moment coefficient about center of gravity
$C_p$	coefficient of pressure
$D$	drag
$d$	base diameter of inner $11.3^\circ$ cone
$E_t$	total energy
$E, F, G$	normal inviscid fluxes at a cell face
$e$	energy
$e, f, g$	normal fluxes in $\xi, \eta, \zeta$ directions, respectively
$\vec{F}$	total surface force vector

$f, g$	functions for creation of cross-sections
$\tilde{f}$	transformation function
$H$	dummy inviscid flux at a cell face
$h$	inviscid flux vectors
$I$	inviscid normal flux
$J$	inviscid flux vector Jacobian
$K$	exponential power
$K$	dummy variable to be linearized
$L$	lift
$l$	length of model
$M$	left eigenvalue matrix
$M^{-1}$	right eigenvalue matrix
$\tilde{M}$	pitching-moment about center of gravity
$\vec{m}$	vector from center of gravity to finite surface area
$n$	unit vector normal to cell face
$\vec{n}$	unit normal to finite surface area
$P$	pressure
$q$	inviscid flux vector
$r$	radius
$s$	minmod function
$t$	time
$U, V, W$	contravariant velocity vectors
$u, v, w$	Cartesian components of velocity
$\vec{V}$	velocity vector
$X$	uniformly spaced coordinate
$x, y, z$	Cartesian coordinates
$Y$	concentrated coordinate

## Greek

$\alpha$	angle of attack
$\alpha_i, \beta_i, \gamma_i$	univariate blending functions
$\gamma$	ratio of specific heats
$\theta$	angle
$\lambda$	matrix containing absolute eigenvalues
$\nu$	distance between cell centers
$\xi, \eta, \zeta$	computational coordinate directions
$\rho$	density
$\Sigma$	total surface area
$\sigma$	cell face area
$\psi$	concentrated coordinate
$\Omega$	cell volume

## Subscripts

a	afterbody, or above body surface
b	below body surface
c	circle, or corrected values on wall
cg	center of gravity
cyl	cylinder
e	ellipse
f	forebody
h	dummy index
i,j,k	indices in $\xi, \eta, \zeta$ directions, respectively
ic	inner cone
n	time level

oc	outer cone
t	time
x,y,z	Cartesian coordinate directions
$\infty$	free-stream values

### **Superscripts**

n	iteration, or time level
T	transpose
*	time level

### **Operators**

$\delta$	difference
----------	------------

# Chapter 1

## INTRODUCTION

Aviation and aerodynamics have developed rapidly since the Wright brothers first flight in 1903. In the past decade, man has gone out into space and returned to a conventional landing on earth (via the space shuttle), spent long durations in space, and sent unmanned space probes beyond our solar system. The next step in the evolution of space travel seems to be the manned exploration of the planets. Mars, because of its close vicinity to earth and its physical characteristics, appears to be the first choice for such a mission. This prompted the study of the atmosphere of Mars in order to determine what type of vehicle and mission could be used to land there. One possibility seems to be a mission in which a vehicle enters the atmosphere, slows to low supersonic speeds, and deploys a parachute. The vehicle can then use its engines to slow down, release the parachute, and land on Mars under its own power.

Many of these types of vehicles were studied in the 1950's and 60's for use in missions to the moon. These vehicles are now being reexamined to see if they may be used in a mission to Mars. One of the vehicles being examined comes from a 1968 technical paper entitled "Aerodynamic Characteristics of a Modified Cone-Conical-Frustum Entry Configuration at Mach 6.0" [1]\*. This paper contains a description of the vehicle and results of experimental tests performed on the vehicle. A review of these results shows that this vehicle might be useful in a manned mission to Mars.

---

\* The numbers in brackets indicate references.

Lift-over-drag ratios are comparable to those needed for entry into the Mars atmosphere at hypersonic and then supersonic speeds.

Methods used to determine the flow characteristics of aircraft and spacecraft have changed a great deal since 1968. Many of the advances made in the areas of aviation and aerodynamics were made possible by the advances made in Computational Fluid Dynamics (CFD). CFD now plays a dominant role in the aerospace field because of its effectiveness as a design tool and as a compliment to experimental tests. The last two decades have brought the introduction of flow solvers which are capable of solving the partial differential equations of fluid motion quickly and efficiently. These solvers have been verified experimentally and are now widely used in the design of aircraft and spacecraft. Continuing improvements in high speed and large memory digital computers also act as a catalyst for the use of CFD in the future. The purpose of this study is to verify the 1968 results and to gather new information about the aerodynamic characteristics of the vehicle.

A typical CFD problem is divided into two steps – grid generation and flow solution. The surface grid in this study is constructed analytically from the original model details [1]. These details, however, lack some information needed to completely define the nose of the vehicle. Therefore, a smooth surface grid is constructed by making some assumptions at the nose and following the original model details as closely as possible.

The volume grid can be constructed using many different methods [2,3]. Some of these include transfinite interpolation, multi-surface interpolation, elliptic grid generation, and hyperbolic grid generation. Transfinite interpolation is used in this study because of the control it allows over the volume grid, its computational efficiency, and the ease in which it can be implemented. Transfinite interpolation requires that the six sides of the volume grid be created before interpolation begins. Two of the six sides are created analytically while the remaining four sides are

generated using an interactive algebraic two-dimensional grid generation algorithm called Two-Boundary Grid Generation (TBGG) [4]. Interpolation is then performed to compute the interior of the volume grid.

Flow solutions can be obtained to varying degrees of accuracy by many different methods. Finite-difference, finite-volume, and finite-element techniques are widely used as a means of solving the partial differential/integral equations of fluid motion. These equations in their viscous, compressible form are the Navier-Stokes equations. Flow solutions obtained from these equations are often computationally expensive. They can, however, be modified in such a way that the diffusion terms are discarded [5]. These modified equations are called the Euler equations. The solution of these equations is computationally more efficient than the solution of the full Navier-Stokes equations and will often produce good estimates of lift and pressure distributions for the vehicle. Values of drag and pitching moment, however, are underestimated, since only form drag and not viscous drag is included.

In this study, the Langley Aerothermodynamic Upwind Relaxation Algorithm (LAURA) [6,7,8] is used to solve for the flowfield around the vehicle. LAURA is a viscous code which has been modified to solve the Euler equations. A literature survey gives a comparison of numerical and experimental results [9] validating the code, and there are a number of examples of the use of the LAURA code for different configurations [10,11].

The experimental tests performed on the vehicle in 1968 [1] were conducted at Mach 6.0. In the present study, computational tests are also performed at Mach 6.0 in order to verify these results and to further describe the flow around the vehicle. A free-stream Mach number of 6.0 falls within the hypersonic range of fluid flow. The characteristics of blunt-nosed bodies in hypersonic flow is well documented [12]. The flow is characterized by the formation of a strong shock wave, subsonic flow on the leading edge of the blunt nose behind the shock, and supersonic or hypersonic

flow behind the shock in the surrounding regions. A compression, overexpansion, and recompression is also expected near the nose in the streamwise direction. Similarly, hypersonic flow around conical bodies is also well documented [12]. It is characterized by strong shock waves coming off the cone at an angle which can be determined analytically based on cone angle and free-stream velocity.

This study is divided into chapters which follow a logical sequence. A detailed description of the generation of the surface and volume grid is given in Chap. 2. The inviscid governing equations for compressible fluid flow are given in Chap. 3. The LAURA code and its modifications to compute inviscid flowfields are also described in this chapter. Chapter 4 gives a summary of the results and comparisons between experimental and computational results. The concluding remarks and the future directions of this study are presented in Chap. 5.

## Chapter 2

# GRID GENERATION

### 2.1 Introduction

The configuration of the spacecraft is taken from a 1968 NASA Technical Note [1]. This design evolved from a basic body of revolution of a  $15.07^\circ$  conical forebody,  $11.3^\circ$  conical afterbody, and a spherical nose. Figures 2.1 and 2.2 show the details of the original model [1]. The afterbody consists of two cylinders mounted in parallel to the exterior of the  $11.3^\circ$  inner cone. These cylinders are tangentially faired into the inner cone as shown in Fig. 2.1. The forebody consists of two  $6.64^\circ$  cones mounted externally to a  $15.07^\circ$  inner cone. These outer cones are also tangentially faired into the inner cone. The spherical nose then fits on the forebody such that  $C^1$  (first derivative) continuity is maintained. Figure 2.3 defines the location of the nose, forebody, and afterbody.

### 2.2 Surface Definition

The surface grid is created analytically from the original model details [1]. Lines of constant  $\xi$  are constructed as lines of constant angle  $\theta$  in the physical space where

$$\theta = \arctan \frac{y}{x} . \quad (2.1)$$

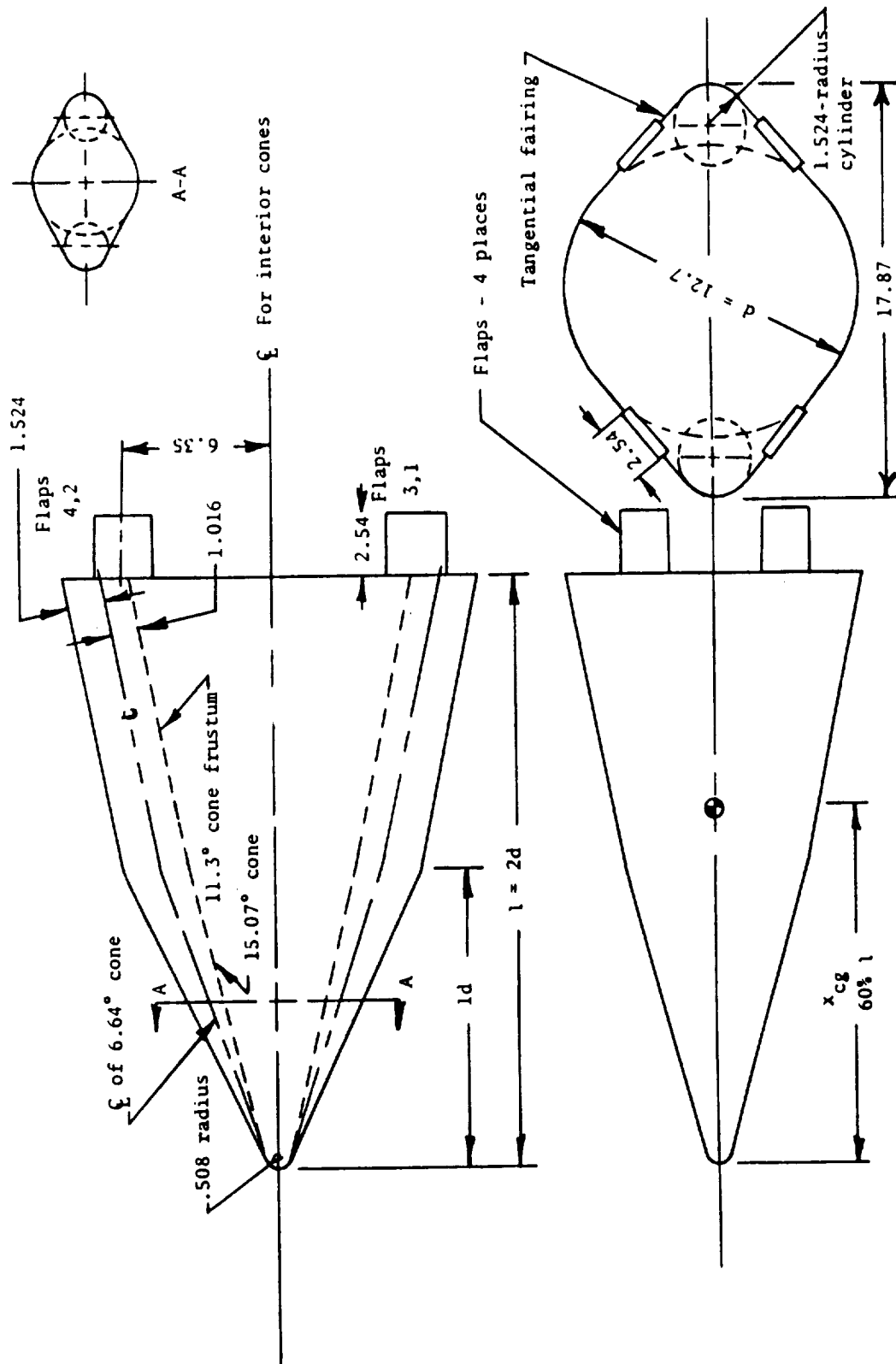


Fig. 2.1 Original model details (all dimensions in cm.).

**Fig. 2.2 Original equipment arrangement.**

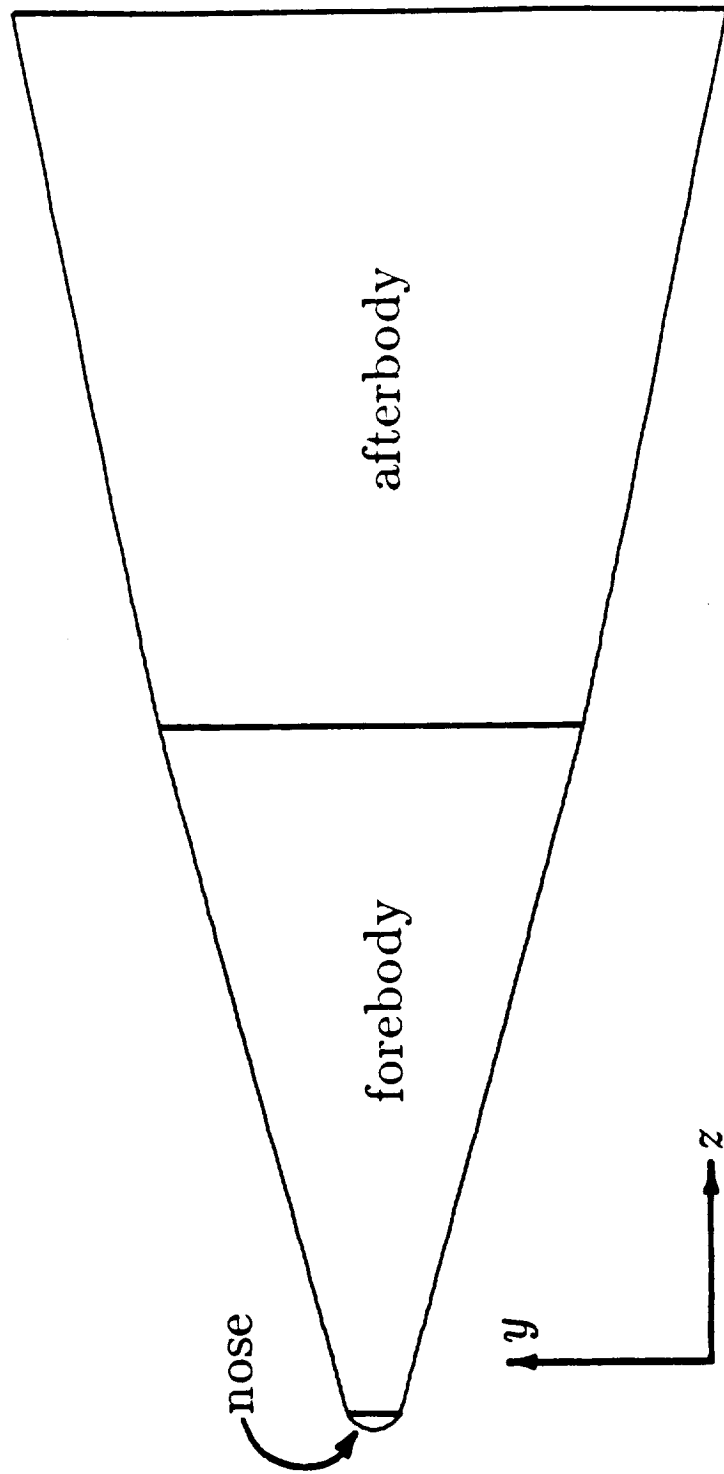


Fig. 2.3 Nose, forebody, and afterbody locations.

Lines of constant  $\eta$  are constructed as lines of constant axial distance  $z$  in the physical space. Therefore,  $\xi$ -lines lie in the  $x$ - $y$  plane and outline cross-sections of the model in a plane perpendicular to the model's axis of symmetry. It is necessary to find the outline of each of these cross-sections in Cartesian coordinates to define the surface of the model.

The afterbody contains cylinders mounted parallel to the walls of an  $11.3^\circ$  inner cone. The cross-sectional cut of a cone perpendicular to its axis of symmetry creates a circle in the  $x$ - $y$  plane (Fig. 2.4a). A cross-sectional cut of a cylinder at  $11.3^\circ$  creates an ellipse (Fig. 2.4c). The minor axis,  $a_a$ , of this ellipse is equal to the radius of the cylinder. The major axis,  $b_a$ , is equal to

$$b_a = \frac{r_{cyl}}{\cos 11.3^\circ} \quad (2.2)$$

The forebody contains two  $6.64^\circ$  cones mounted at an angle of  $15.07^\circ$  to the axis of the inner cone. A cross-sectional cut of these cones at an angle of  $15.07^\circ$  creates an "egg-shaped" ellipse (Fig. 2.4b). The length of the major axis on one side of the ellipse is different than that of the other. Noting that the radius of the inner cone is always larger than that of the outer cone for each cross-section (Fig. 2.1), the innermost major axis is neglected. The tangent between the inner cone and outer cone will always intersect the outermost side of the egg-shaped ellipse. The outer major axis is calculated as

$$b_f = \frac{r_{oc}}{\cos 15.07^\circ} \quad (2.3)$$

The radius  $r_{oc}$  is the radius of the outer cone at the point of intersection of the outer cone axis and the cross-sectional plane (Fig. 2.4b). The minor axis,  $a_f$ , is equal to the radius of the outer cone at this point. The spherical nose begins at the point where the outer cone reaches its apex. A cross-section of the spherical nose creates a circle in the  $x$ - $y$  plane.

Figure 2.5 represents a typical cross-section of a region in the forebody or afterbody. The radius, major axis, minor axis, and ellipse center are now determined

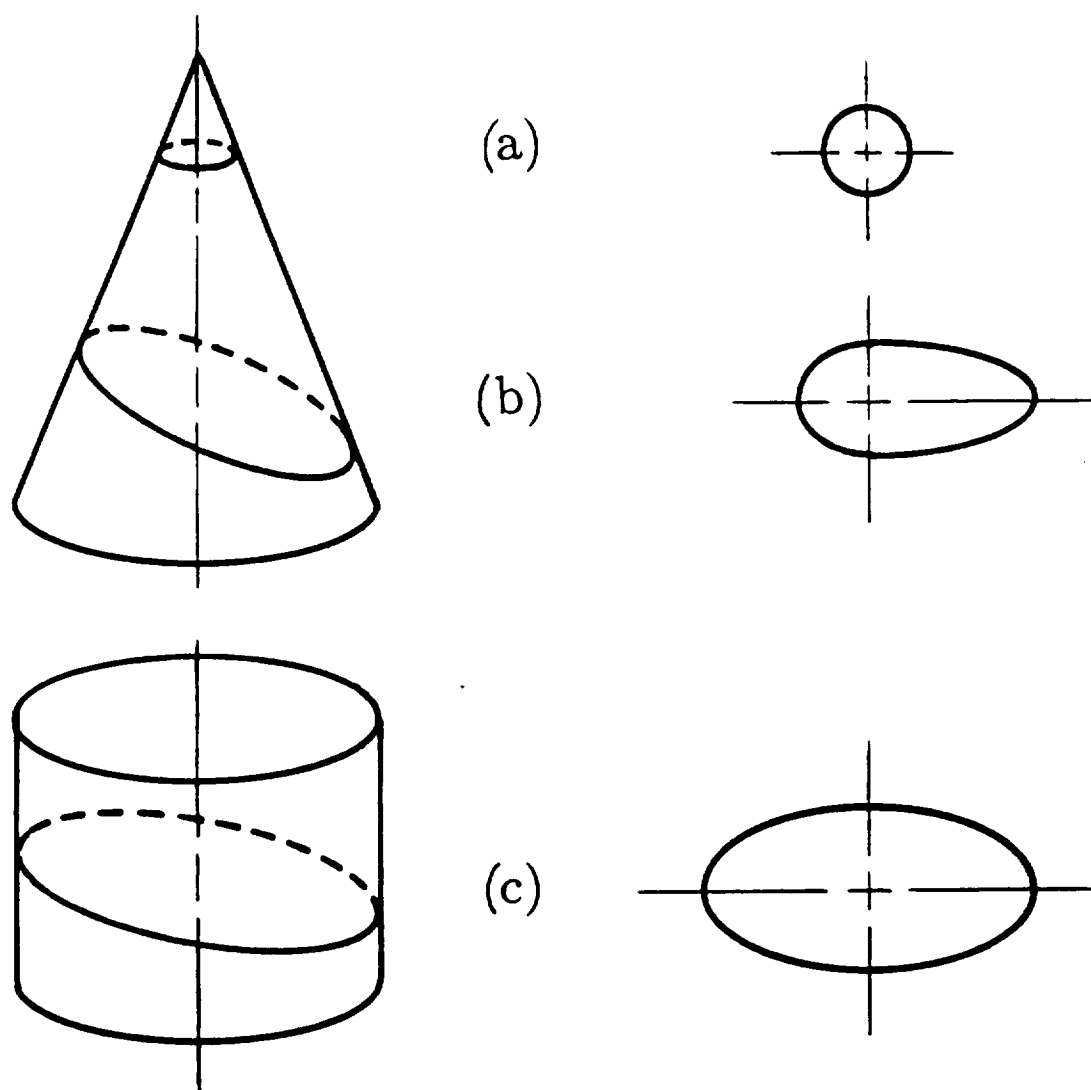


Fig. 2.4 Cross-sections of the vehicle: (a)circle, (b)egg-shaped ellipse, and (c)ellipse.

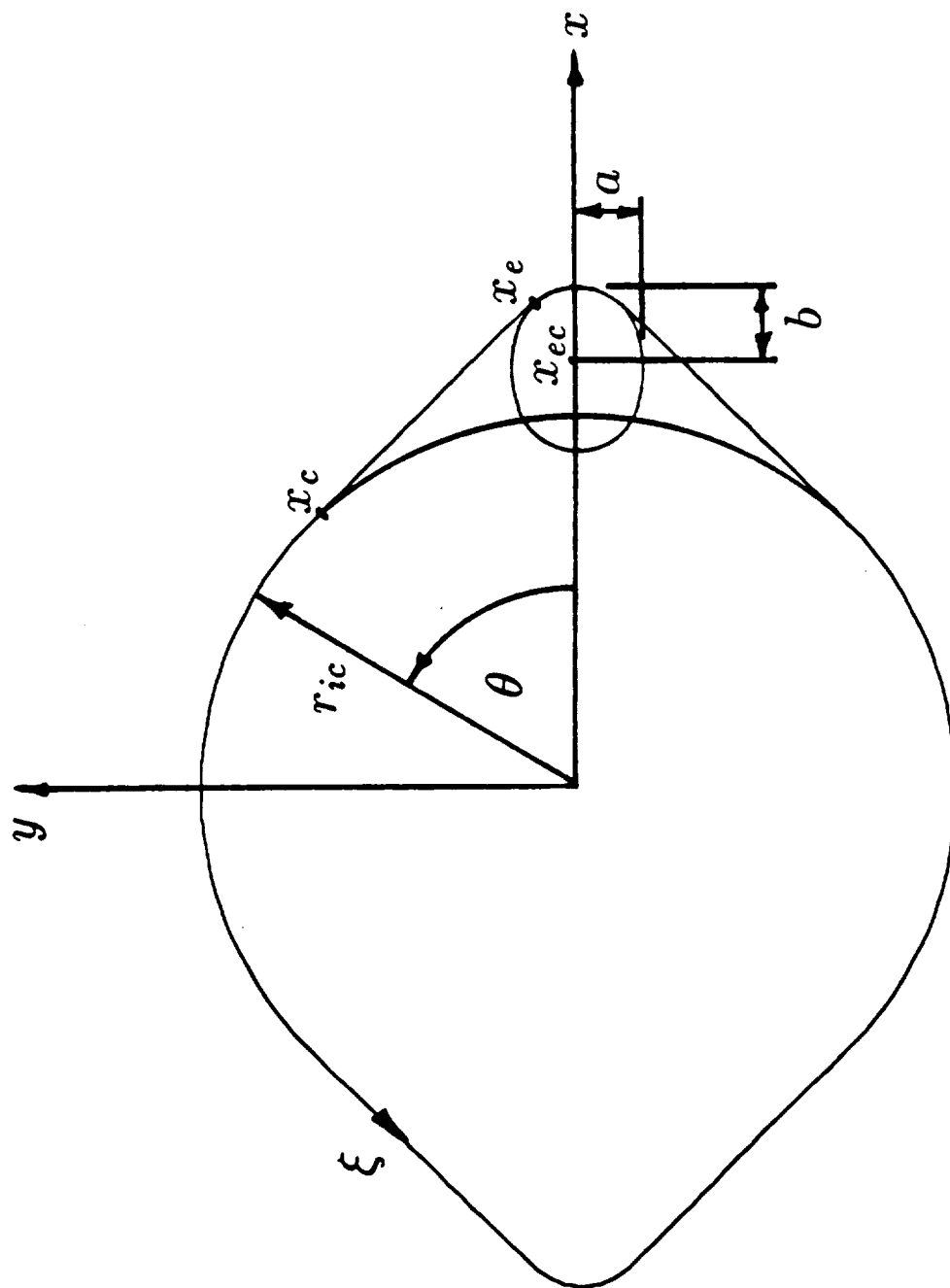


Fig. 2.5 Forebody or afterbody cross-section.

from the information given above. This information is necessary in order to determine the points of tangency on the circle and ellipse. These points, along with the first derivatives at these points, represent six unknowns. Six equations can be constructed to determine these unknowns. The equations are:

Equation of a circle

$$x_c^2 + y_c^2 = r_{ic}^2 \quad (2.4)$$

Equation of an ellipse

$$\left(\frac{x_e - x_{ec}}{b}\right)^2 + \left(\frac{y_e}{a}\right)^2 = 1 \quad (2.5)$$

Equation of a triangle

$$r_{ic}^2 + [(x_c - x_e)^2 + (y_c - y_e)^2] = [x_e^2 + y_e^2] \quad (2.6)$$

Equation of a tangent at  $(x_c, y_c)$

$$\frac{\partial y_c}{\partial x_c} = -\frac{x_c}{y_c} \quad (2.7)$$

Equation of a tangent at  $(x_e, y_e)$

$$\frac{\partial y_e}{\partial x_e} = -\left(\frac{x_e - x_{ec}}{y_e}\right) \frac{a^2}{b^2} \quad (2.8)$$

Equal tangents equation

$$\frac{\partial y_c}{\partial x_c} = \frac{\partial y_e}{\partial x_e} \quad (2.9)$$

Equations (2.4 - 2.9) can be simplified into two equations

$$f(x_c, x_e) = \frac{x_e - x_{ec}}{\sqrt{1 - [(x_e - x_{ec})/b]^2}} \frac{a}{b^2} - \frac{x_c}{\sqrt{r_{ic}^2 - x_c^2}} = 0 \quad , \quad (2.10)$$

$$g(x_c, x_e) = 2r_{ic}^2 - 2x_c x_e - 2a \left\{ \left[ 1 - \left( \frac{x_e - x_{ec}}{b} \right)^2 \right] [r_{ic}^2 - x_c^2] \right\}^{1/2} = 0 \quad , \quad (2.11)$$

where  $b$  is the major axis of the ellipse at each cross-section ( $b_a$  or  $b_f$ ), and  $a$  is the minor axis of the ellipse at each cross-section ( $a_a$  or  $a_f$ ). Equations (2.10) and (2.11) can be combined into one non-linear equation. It is more convenient, however, to

leave them as separate equations and solve for  $x_c$  and  $x_e$  using a Newton-Raphson method. The matrix form of this method is

$$\begin{bmatrix} f_{x_c} & f_{x_e} \\ g_{x_c} & g_{x_e} \end{bmatrix} \begin{bmatrix} \Delta x_c \\ \Delta x_e \end{bmatrix} = \begin{bmatrix} -f \\ -g \end{bmatrix} \quad , \quad (2.12)$$

where  $\Delta x_c = x_c^{n+1} - x_c^n$ ,

$\Delta x_e = x_e^{n+1} - x_e^n$ ,

and subscripts  $x_c$  and  $x_e$  denote differentiation with respect to these variables. Initial values  $x_c^n$  and  $x_e^n$  are required to begin the solution of Eq. (2.12). These values can be found by simply estimating the points of tangency on the circle and ellipse. These estimations, however, can be inaccurate and in some cases result in the divergence of the solution. To avoid this problem, a new method is developed to find the initial values. As noted earlier, the point of tangency on the ellipse is always on the side away from the inner cone. In other words,  $x_e$  is always between  $x_{ec}$  and  $x_{ec} + b$ . This allows an initial value of  $x_{ec} + .5b$  to be used for  $x_e^n$ . The initial value  $x_c^n$  is then found by solving Eq. (2.10) for  $x_c$  as a function of  $x_e$ , i.e.

$$x_c(x_e) = \left\{ \frac{\left[ (x_e - x_{ec})^2 a^2 r_{ic}^2 / \left( 1 - \left( \frac{x_e - x_{ec}}{b} \right)^2 \right) b^4 \right]}{\left\{ 1 + \left[ (x_e - x_{ec})^2 a^2 / \left( 1 - \left( \frac{x_e - x_{ec}}{b} \right)^2 \right) b^4 \right] \right\}} \right\}^{1/2} \quad . \quad (2.13)$$

The Newton-Raphson method works well in finding  $x_c$  and  $x_e$  over most of the model. Near the nose, however, the solution fails to converge. The bisection method is then used to determine  $x_c$  and  $x_e$  over the remaining cross-sections. This method requires that there be only one unknown. Equation (2.13) is therefore used to find  $x_c$  as a function of  $x_e$ . This equation is then substituted into Eq. (2.11) such that

$$g = g(x_e) \quad . \quad (2.14)$$

The solution is initially bracketed by  $x_{ec}$  and  $x_{ec} + b$ . The value of  $x_e$  determined from this method is then substituted into Eq. (2.13) to determine  $x_c$ .

The coordinate lines  $\xi$  and  $\eta$  can now be constructed on the surface of the model in the physical space. The y-coordinates corresponding to  $x_c$  and  $x_e$  are found from Eqs. (2.4) and (2.5), respectively. These equations, along with the equation of a line between  $(x_c, y_c)$  and  $(x_e, y_e)$ , are used to create the coordinate line  $\xi$  in the physical space. Coordinate line  $\eta$  is constructed using Eq. (2.1), noting that each  $\eta$  line is at a constant angle  $\theta$  in the physical space.

A side and top view of the surface grid is given in Fig. 2.6. These views show concentrations of grid lines at certain locations on the model. The concentrations are necessary in order to completely resolve possible shocks in these regions. This will be discussed in the following chapter. If  $X$  is a uniformly spaced coordinate, the equation

$$Y = \frac{e^{KX} - 1}{e^K - 1}, \quad (2.15)$$

where  $0 \leq Y \leq 1$ ,

$$0 \leq X \leq 1,$$

is used to concentrate grid points near  $Y=0$  for  $K > 0$ , or near  $Y=1$  for  $K < 0$ . High concentrations of grid points result when large positive or negative values of  $K$  are used. Successive concentrations are obtained by connecting a number of curves created by Eq. (2.15) such that they are  $C^1$  continuous.

## 2.3 Nose Discontinuity

The construction of the afterbody is fairly straightforward. Figure 2.1 gives all the information necessary for its construction. The information on the forebody and nose, however, is incomplete. Specifically, more information is needed to determine how the spherical nose fits onto the forebody. Since this information is unavailable, some assumptions are made.

The initial assumption is that each outer cone reaches its apex at the same axial location that the inner cone is truncated (Fig. 2.7). A cross-section of the

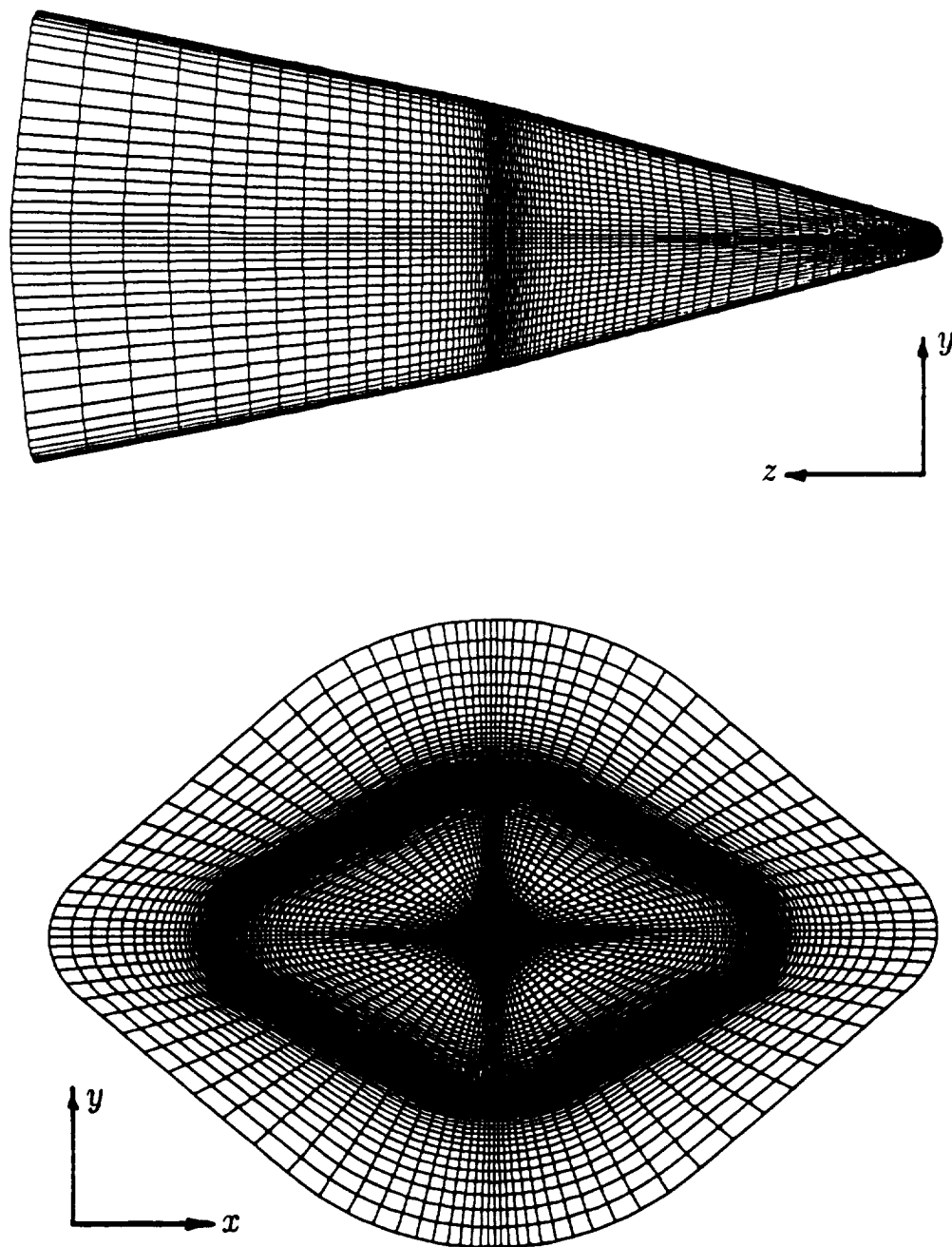


Fig. 2.6 Side and top view of surface grid.

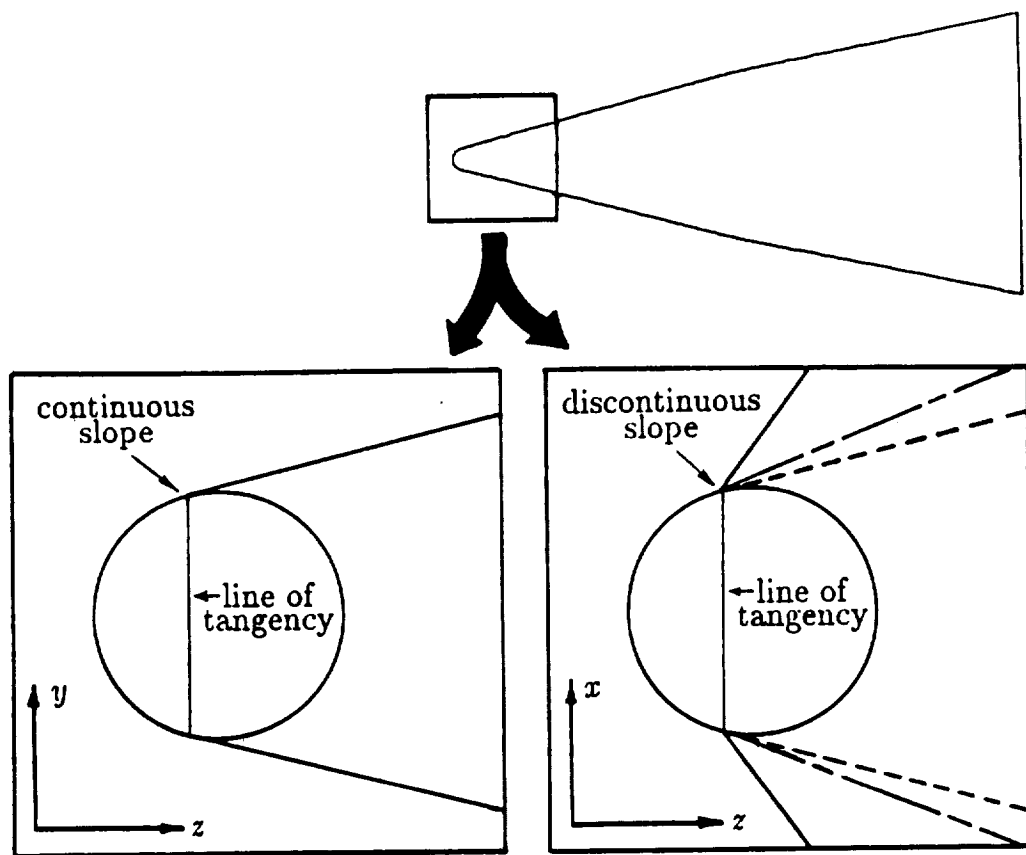


Fig. 2.7 Initial nose configuration.

model taken perpendicular to the model's axis of symmetry at this point will result in a circle. Theoretically, a spherical cap can then fit onto the model. Upon close examination of the model, however, this method results in a spherical nose that has a point discontinuity in slope at the apex of each outer cone. The outer cones approach the spherical cap at an angle of  $25.59^\circ$  to the model's axis of symmetry while the inner cone approaches at an angle of  $15.07^\circ$  (Fig. 2.7). If the spherical cap is made  $C^1$  continuous on the inner cone, then it will not be  $C^1$  continuous at each outer cone's apex and vice versa.

A logical choice is to make the spherical cap  $C^1$  continuous on the inner cone and to smooth out the two point discontinuities in slope caused by the outer cones. A smoothing algorithm is used to smooth these points locally. Details of this algorithm can be found in Refs. 11 and 13. The algorithm creates piecewise continuous cubic splines through discontinuous regions allowing the user to smooth particular regions of the model without affecting the surrounding regions. When applied to the model, the program essentially "fills in" the step change in slope at each discontinuity (Fig. 2.8). This, however, creates a new problem. The regions adjacent to the smoothed region are now "lower" than the smoothed region as a result of the program filling in the discontinuity. This causes "valleys" to appear in the surface. These valleys cannot remain on the finished model. They can essentially be flattened out by smoothing a larger region of the model. This, however, is not done since the original model configuration is lost when large regions are smoothed. If a reasonable model is to be created, it will be necessary to reexamine some of the original assumptions made at the nose.

The assumption that the outer cones reach their apex at the same axial location that the inner cone is truncated, is modified slightly to allow a nose to be created that does not require smoothing. This is done by keeping the spherical cap  $C^1$  continuous on the inner cone, but allowing each outer cone's apex to intersect

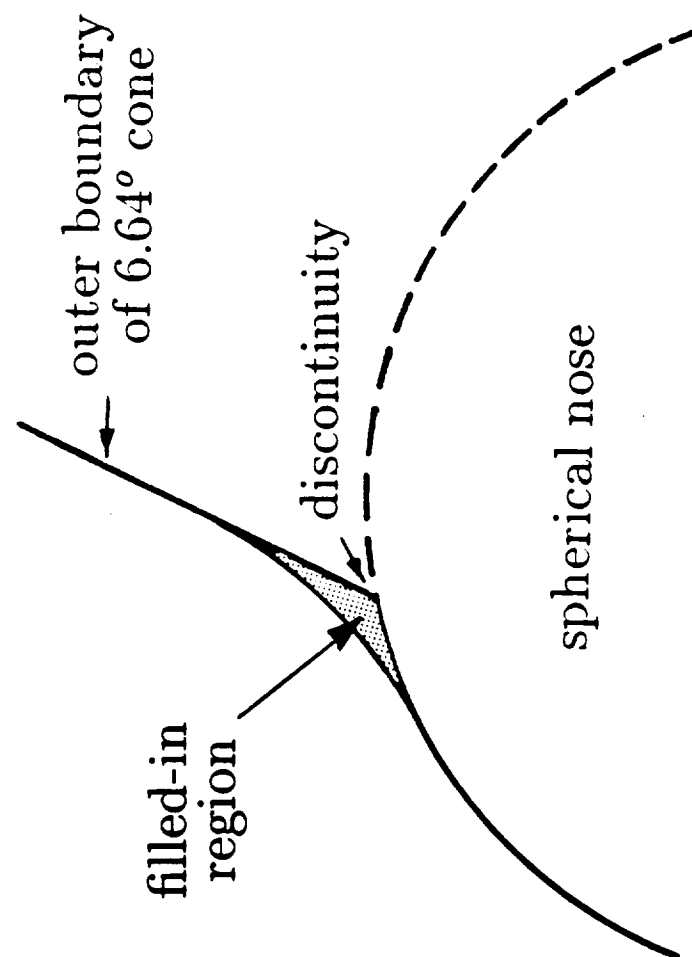


Fig. 2.8 Filling in discontinuity.

the spherical cap such that they are  $C^1$  continuous with the cap (Fig. 2.9). This essentially moves each outer cone towards the spherical cap slightly, allowing their apexes to intersect the cap tangentially. In the small region where the outer cones extend beyond the forebody, the outer cones are tangentially faired into the spherical nose rather than the inner cone. This requires that the inner cone radius,  $r_{ic}$ , be replaced by the spherical nose radius,  $r_n$ , in Eqs. (2.4) and (2.6), and subsequently in Eqs. (2.10) and (2.11). These equations are then solved using the same methods described in Sec. 2.2.

The changes imposed on the nose of the model have an effect on the forebody-afterbody junction of the spacecraft. The  $11.3^\circ$  inner cone of the afterbody and the  $15.07^\circ$  inner cone of the forebody are still  $C^0$  continuous. No changes are made on either inner cone when the nose is changed. The outer cones of the forebody and the outer cylinders of the afterbody, however, are no longer  $C^0$  continuous. If the locations of the apex of each outer cone is changed, the location of the entire cone is subsequently changed. The  $C^0$  continuity is maintained at the junction if the  $6.64^\circ$  outer cones are replaced with cones of frustum angles less than  $6.64^\circ$ . This, however, is not done since the outer cones of the forebody are explicitly given to be  $6.64^\circ$  in the original model details (Fig. 2.1). Instead, the angle of inclination of the outer cones is changed to accommodate the changes made in the nose.

Figure 2.10 shows the new configuration of the forebody-afterbody junction. The difference in radius between the base of the outer cones and the outer cylinders causes a small  $C^0$  discontinuity at the junction but this can easily be smoothed using the smoothing algorithm described earlier [13]. Figure 2.11 shows the discontinuity before and after smoothing. Figure 2.12 gives a detailed schematic of the new model details and the two programs which generate the surface grid are given in Appendix A.

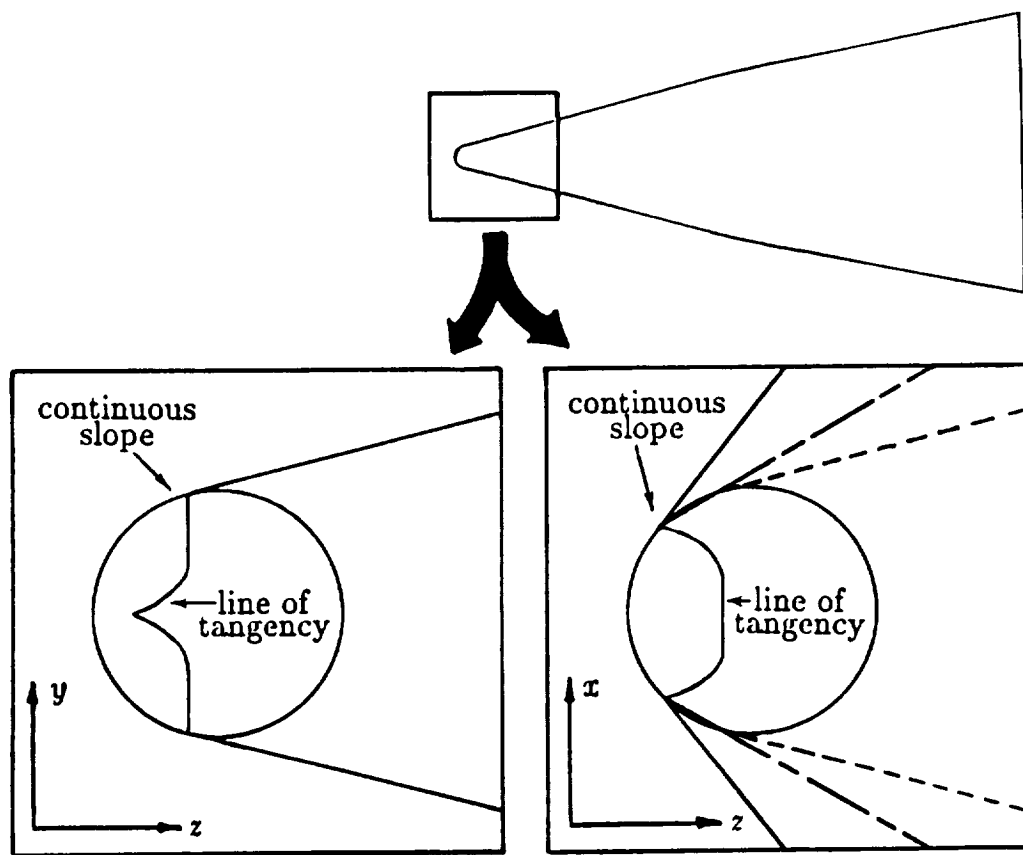


Fig. 2.9 Final nose configuration.

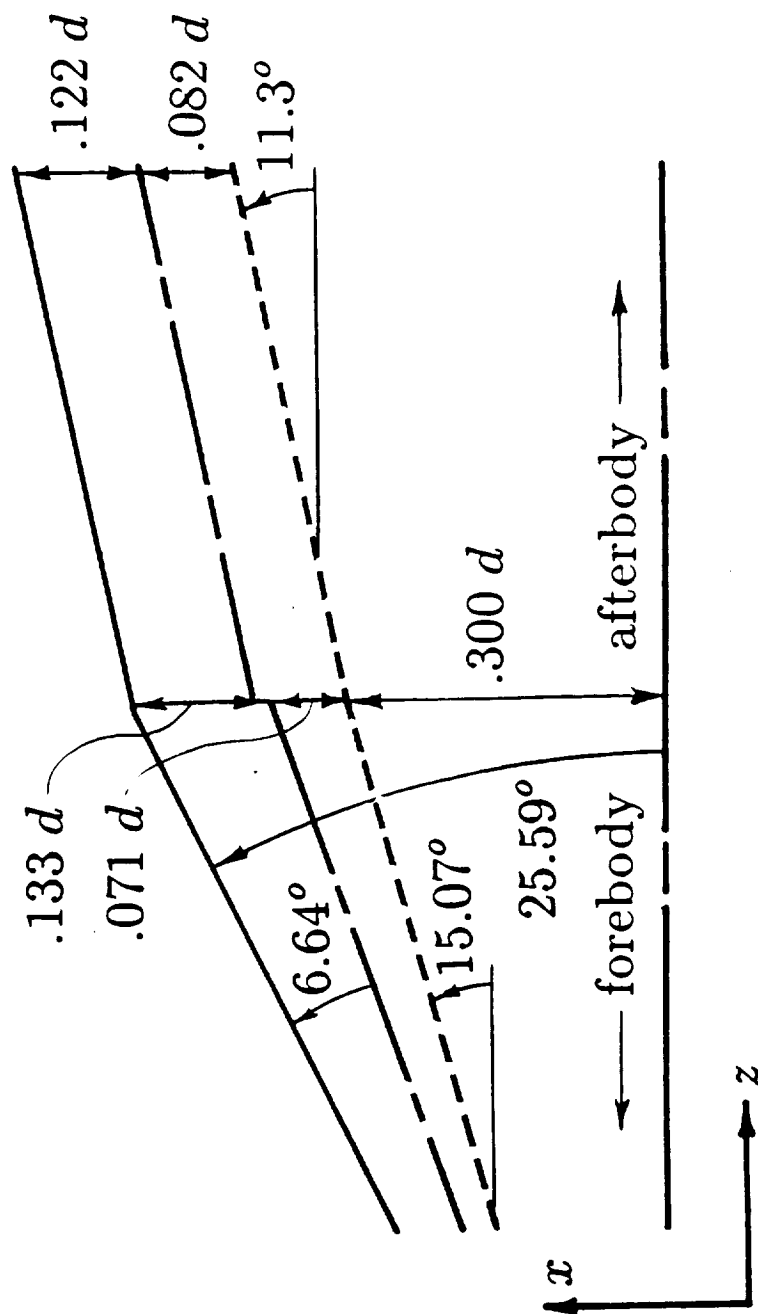


Fig. 2.10 Forebody/afterbody junction.

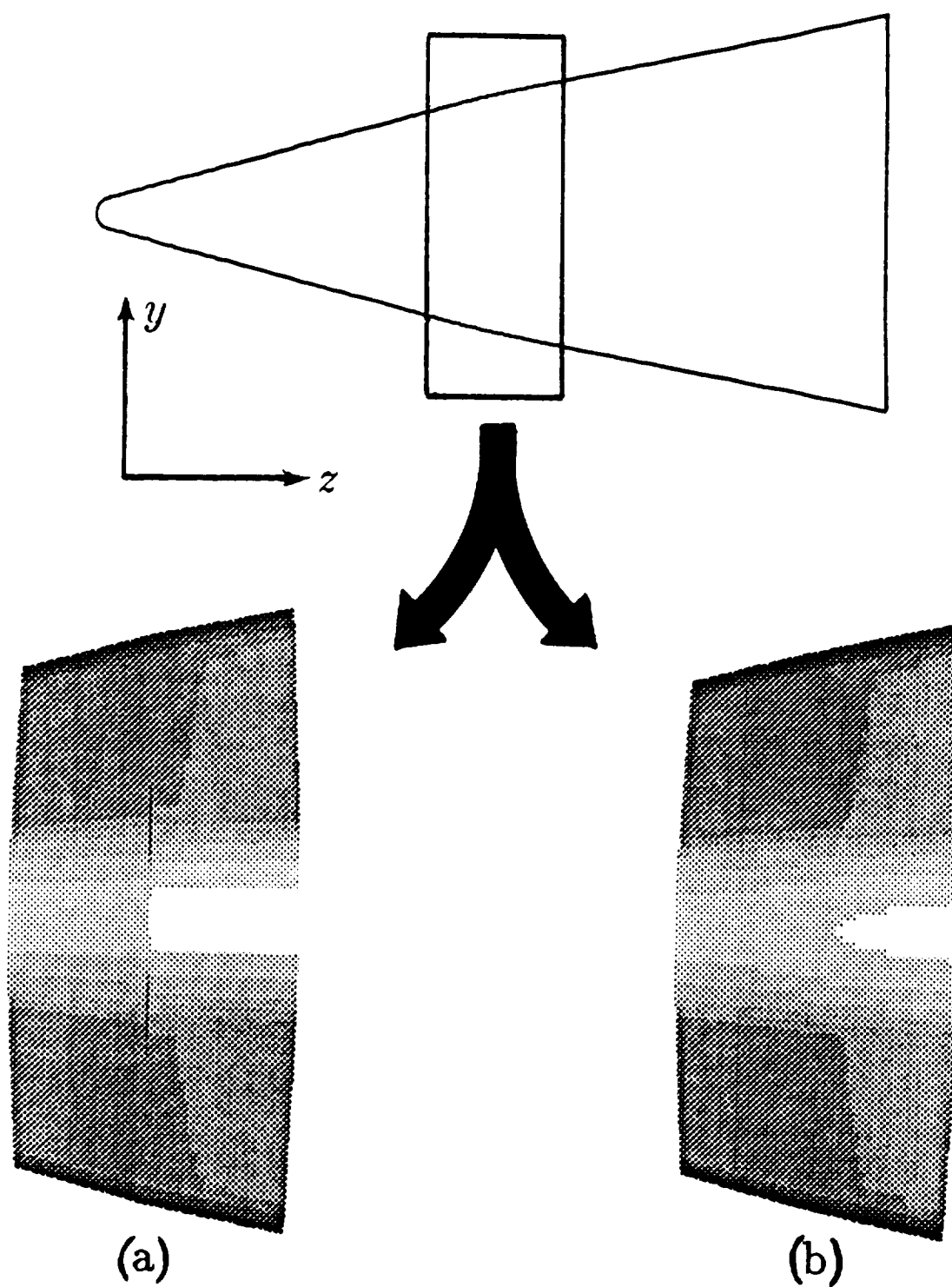


Fig. 2.11 Surface discontinuity: (a) before smoothing, and (b) after smoothing.

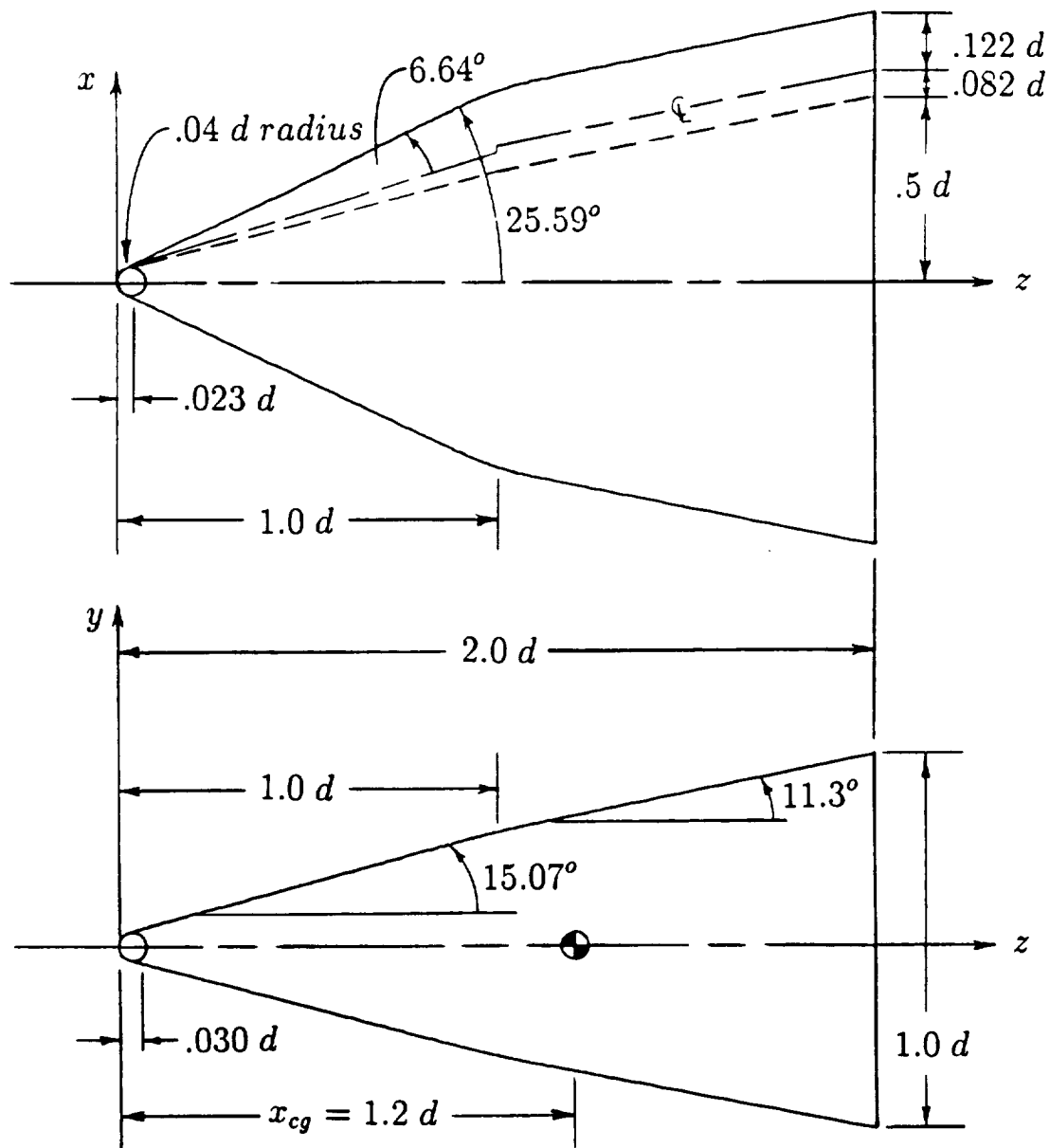


Fig. 2.12 New model details.

## 2.4 Volume Grid

A single-block volume grid can now be generated about the surface grid. The volume grid is a C-O type grid with coordinate direction  $\zeta$  defined normal to both  $\xi$  and  $\eta$  in the computational domain. Flow solutions for the vehicle are obtained at various angles of attack but at  $0^\circ$  yaw. It is therefore only necessary to generate a volume grid around half of the vehicle. The vehicle enters the atmosphere with the x-axis (Fig. 2.12) oriented horizontally. The vehicle is therefore divided along the y-z-axis and a grid is generated around one half (side view of Fig 2.6).

The volume grid is generated in two steps. First, the top symmetry, bottom symmetry, side, exit, and outer grids are created. Then the interior of the volume grid is generated using transfinite interpolation. The top symmetry, bottom symmetry, side, and exit grids are created using an interactive, algebraic, two-dimensional grid generation program called TBGG (Two-Boundary Grid Generation). This program is described by Smith and Wiese in a 1986 NASA technical paper [4]. TBGG first reads an input data file containing points which define the four outer boundaries of the 2-D grid. The points on the first boundary of each 2-D grid are defined by the existing surface grid. These points are fixed in TBGG so that they always coincide with the points on the surface grid. The three remaining boundaries of the 2-D grids are determined in such a way as to capture the shocks coming off the vehicle.

The top, bottom, and side grids share one common boundary. This boundary is along the negative z-axis and is shared by all 2-D grids in the  $\eta$ - $\zeta$  computational plane. This boundary is a polar singularity and lies along the stagnation line of the vehicle at  $0^\circ$  angle of attack. The length of this line should be at least 1.5 times the shock stand-off distance expected in front of the blunt nose in order to properly capture the shock. At Mach 6.0 the shock stand-off distance is approximately .006  $d$  ( $d$  being the base diameter of the  $11.3^\circ$  inner cone shown in Fig. 2.12) for a .08  $d$  diameter sphere [14]. It is very difficult to create a reasonable grid within a .009  $d$

region between the blunt nose and the volume grid's outer boundary. The grid lines coming off the surface would require a high degree of curvature in order to maintain orthogonality at the surface. Therefore, the outer boundary is moved away from the blunt nose a distance of three times the diameter of the spherical nose. The increased distance allows a more uniform grid to be created in front of the nose.

The two remaining boundaries of the top, bottom, and side grids are determined from the shock wave angle created by a cone. The shock wave angle for a  $15.07^\circ$  cone at Mach 6.0 is approximately  $19^\circ$  while the shock angle for a  $25.59^\circ$  cone is approximately  $30^\circ$  [15]. Therefore, in order to capture the shock wave around the vehicle, the outer boundaries of the top and bottom grids are inclined to an angle of  $25^\circ$  while the outer boundary of the side grid is inclined to an angle of  $35^\circ$ . If computations are performed at different angles of attack, the outer boundary of the top grid must be inclined to an angle greater than  $25^\circ$  in order to properly capture the shock. Estimating that the shock wave angle for a  $15.07^\circ$  cone at an angle of attack of  $15^\circ$  is equal to the sum of the shock wave angle (at  $0^\circ$  angle of attack) and the angle of attack, an outer boundary inclined at  $40^\circ$  will capture the shock. Therefore, this outer boundary captures all shocks for angles of attack up to  $15^\circ$ . Using this information, the four boundaries of the top, bottom, and side grids can be generated for input into TBGG.

The outer boundaries of the top, bottom, and side grids are used in the creation of the boundaries of the exit grid. The top, bottom, and surface grids coincide with the exit grid on three of the four boundaries. The point distributions on these boundaries are fixed in TBGG so that they will always coincide. The remaining boundary is defined by the intersection of the top, bottom, and side grid with the exit grid. This boundary is essentially a "stretched" circle. The radius of the circle is determined by linear interpolation between intersections of each grid with the exit grid.

TBGG begins with the outer boundaries of each 2-D grid. The user then specifies the number of grid points, point distributions (assuming they are not already defined as described above), orthogonality, etc. in order to generate a 2-D grid. The grid is then viewed and modified until a satisfactory grid is created. Point concentrations are made at the nose, near the vehicle, and near the forebody/afterbody junction in order to resolve high gradients expected in these regions. The resulting 2-D top, bottom, side, and exit grids are shown in Fig. 2.13.

The next step in generating the volume grid is to create the sixth side of this grid. The polar singularity, surface, top, bottom, and exit grids are the five sides of the volume grid which now exist. The side grid is not a side of the volume grid but an intermediate grid between the top and bottom grids. It is used to control the shape and point distributions of the volume grid in the physical space. The final side of the volume grid is called the "cap". The cap is a 3-D surface in the physical space but exists as a 2-D surface in the computational space. This grid is in the  $\xi$ - $\eta$  computational plane opposite the surface grid. Two boundaries of the cap are defined by their intersection with the top and bottom grids. The third boundary of the cap is defined by the location of the polar singularity while the fourth boundary is defined by the exit grid. All of these boundaries have point distributions which are defined in the physical space (excluding the polar singularity which is a point). It is necessary to define the interior of the cap before generating the interior of the volume grid.

TBGG is not used to generate the interior grid of the cap since the cap is a surface in the physical space. Instead, the interior of the cap is generated based on the point distributions given on its boundaries and on the point distributions given on the side grid. The side grid intersects the cap along an  $\eta$ -coordinate line lying between the top and bottom grid boundaries. The points on this line are used to control the shape of the cap which in turn controls the shape of the volume grid. The cap is generated by first considering cross-sections of this grid projected on the x-y plane. Each coordinate

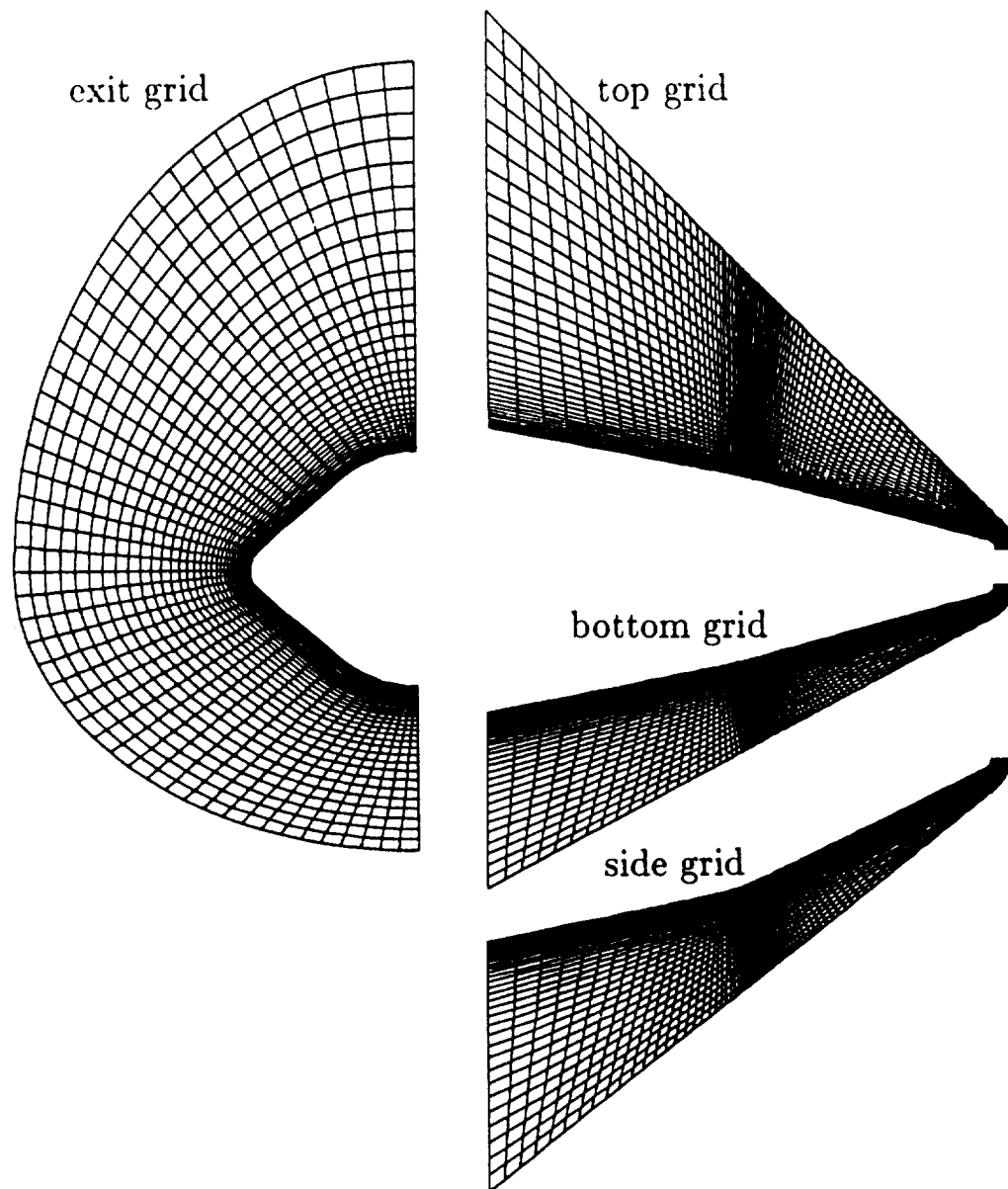


Fig. 2.13 Top, bottom, side, and exit grids.

line  $\xi$  of the cap is generated in the same way that the outer boundary of the exit grid is generated. Three points are defined on this line by intersections of the cap with the top, bottom, and side grids. Linear interpolation between these points is used to find the radius of a stretched circle in the x-y plane. The circle is generated through these points to create each  $\xi$ -coordinate line. The point distributions along the  $\xi$  coordinate lines are found based on the angular distributions of the points along the joint boundary of the cap and the exit grid. Therefore, each coordinate line  $\eta$  lies at a constant angle  $\theta$ , where  $\theta$  is defined by Eq. (2.1). Figure 2.14a shows the shape of a typical cap cross-section projected on the x-y plane.

The x and y locations of the cap's interior are now defined in the physical space. The z locations of the grid are found by looking at the grid projected on the y-z physical plane (Fig. 2.14b). Three points are defined on the  $\xi$  coordinate line by the intersections of the top, bottom, and side grids with the cap. A  $C^1$  continuous line must be constructed through these points in order to define the z locations of the  $\xi$ -coordinate lines in the physical space. This is done by creating a quadratic equation of the form

$$z = a_1 y^2 + a_2 y + a_3 \quad , \quad (2.16)$$

where  $a_1$ ,  $a_2$ , and  $a_3$  are coefficients found by solving Eq. (2.16) simultaneously through the three points. This equation, therefore, creates a quadratic curve which is  $C^1$  continuous through these points. Since the y locations of each coordinate line  $\eta$  are known, the z locations of these coordinate lines are also found from Eq. (2.16). The interior of the cap is therefore completely defined in the physical space and is shown in Fig. 2.15.

Transfinite interpolation is now used to generate the volume grid. The theory of transfinite interpolation as described by Gordon and Hall [16] is a general concept of multivariate interpolation between any number of surfaces. The single-block volume grid needed for this vehicle requires interpolation between the two

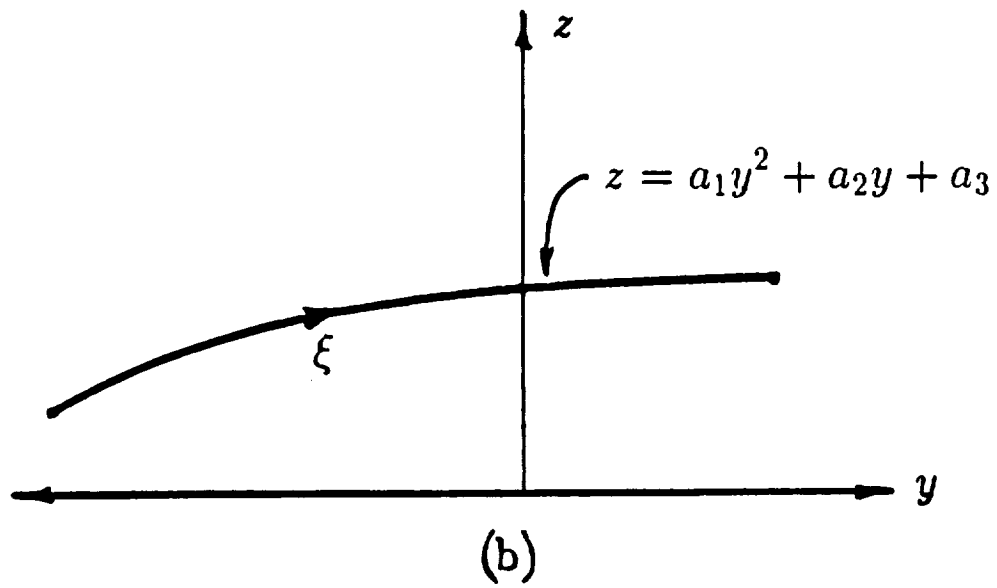
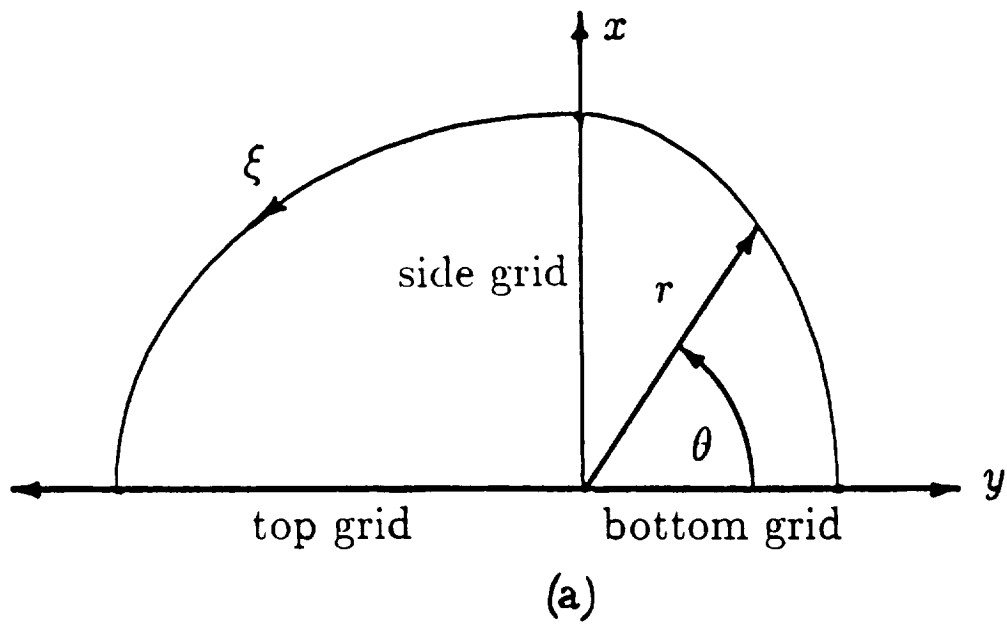


Fig. 2.14 Definition of cap: (a)x-y plane, and (b)y-z plane.

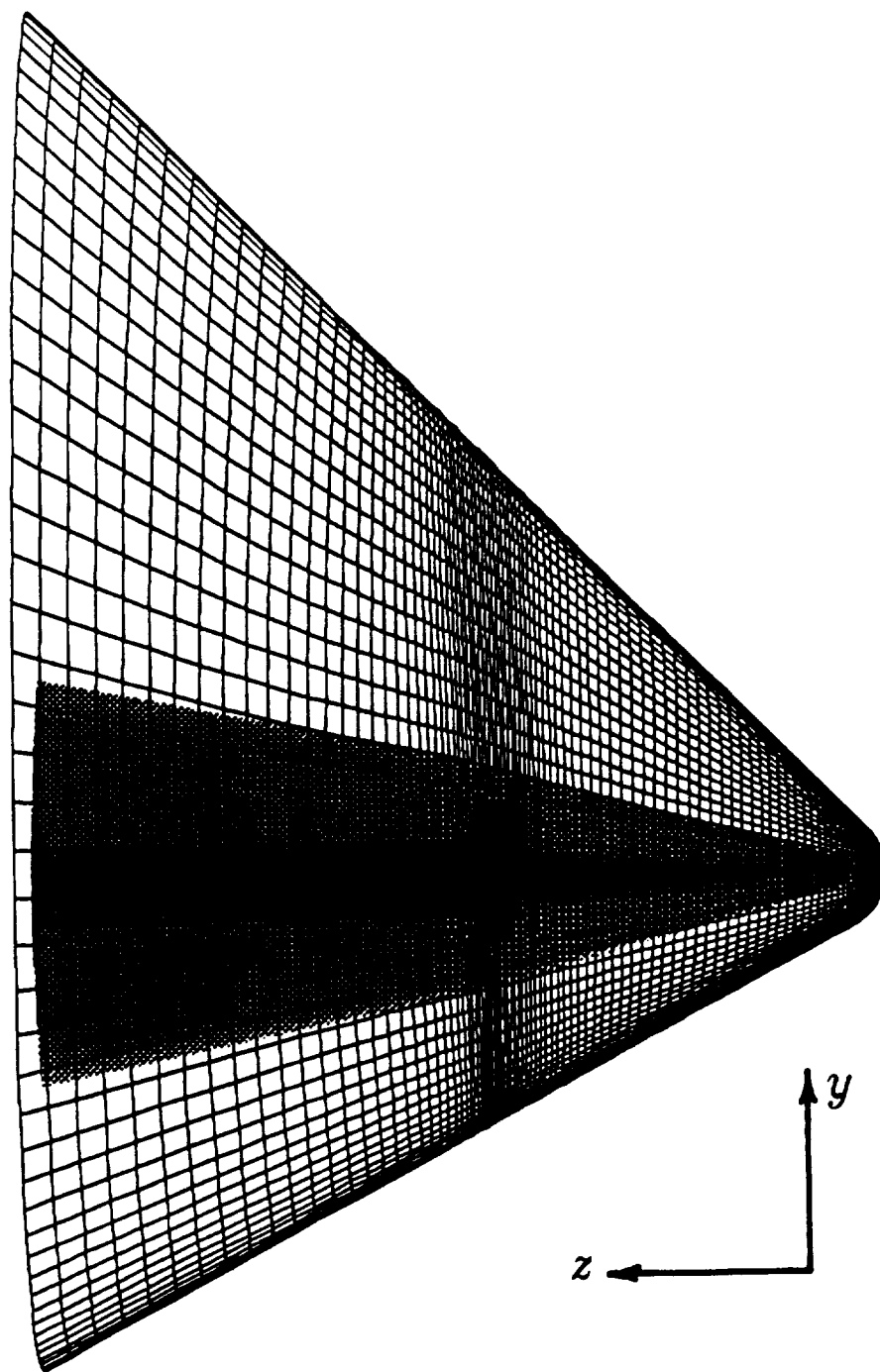


Fig. 2.15 Definition of cap in the physical space.

outermost planes of the  $\xi$ ,  $\eta$ , and  $\zeta$  coordinate directions. The transformation function  $\vec{f}(\xi, \eta, \zeta)$  defines the transformation from the unit cube computational domain to an arbitrarily-shaped region in the physical domain. Using this definition, the transfinite interpolation procedure is a recursive algorithm which is used to generate the interior of the volume grid. This algorithm is given as

$$\begin{aligned}\vec{f}(\xi, \eta, \zeta) &= [x(\xi, \eta, \zeta), y(\xi, \eta, \zeta), z(\xi, \eta, \zeta)]^T, \\ \vec{f}_1(\xi, \eta, \zeta) &= \sum_{h=1}^2 \alpha_h(\xi) \vec{a}_h(\eta, \zeta), \\ \vec{f}_2(\xi, \eta, \zeta) &= \vec{f}_1(\xi, \eta, \zeta) + \sum_{h=1}^2 \beta_h(\eta) [\vec{b}_h(\xi, \zeta) - \vec{f}_1(\xi, \eta_h, \zeta)], \\ \vec{f}(\xi, \eta, \zeta) &= \vec{f}_2(\xi, \eta, \zeta) + \sum_{h=1}^2 \gamma_h(\zeta) [\vec{c}_h(\xi, \eta) - \vec{f}_2(\xi, \eta, \zeta_h)].\end{aligned}\quad (2.17)$$

The planes  $\vec{a}$ ,  $\vec{b}$ , and  $\vec{c}$  are defined as

$$\begin{aligned}\vec{a}_1(\eta, \zeta) &= \vec{f}(0, \eta, \zeta), & \vec{a}_2(\eta, \zeta) &= \vec{f}(1, \eta, \zeta), \\ \vec{b}_1(\xi, \zeta) &= \vec{f}(\xi, 0, \zeta), & \vec{b}_2(\xi, \zeta) &= \vec{f}(\xi, 1, \zeta), \\ \vec{c}_1(\xi, \eta) &= \vec{f}(\xi, \eta, 0), & \vec{c}_2(\xi, \eta) &= \vec{f}(\xi, \eta, 1).\end{aligned}\quad (2.18)$$

The univariate blending functions  $\alpha$ ,  $\beta$ , and  $\gamma$  are of the form

$$\begin{aligned}\alpha_1(\xi) &= 1 - \psi(\xi), & \alpha_2(\xi) &= \psi(\xi), \\ \beta_1(\eta) &= 1 - \psi(\eta), & \beta_2(\eta) &= \psi(\eta), \\ \gamma_1(\zeta) &= 1 - \psi(\zeta), & \gamma_2(\zeta) &= \psi(\zeta),\end{aligned}\quad (2.19)$$

where  $\psi(X) = \frac{e^{KX} - 1}{e^K - 1}$ ,  $0 \leq X \leq 1$ .

The exponential equation above is identical to Eq. (2.15) of Sec. 2.2. The effects of different values of  $K$  on  $\psi$  are described in that section.

Transfinite interpolation is performed on the volume grid in two steps. First, interpolation is performed between the top and side grids with the top grid defined as  $\vec{a}_1$  and the side grid defined as  $\vec{a}_2$ . Then, interpolation is performed between the

side and bottom grids with the bottom grid defined as  $\vec{a}_1$  and the side grid defined as  $\vec{a}_2$ .  $C^1$  continuity cannot be guaranteed across the side grid boundary if the volume grid is generated in this manner but visual inspection has shown that a high degree of continuity is maintained across this boundary.

A value of 2 is chosen for  $K$  in Eq. (2.19). Noting the definitions of  $\alpha$ ,  $\beta$ , and  $\gamma$  in Eq. (2.19), and the definition of the transformation function in Eq. (2.17), the majority of the emphasis is placed on the  $\vec{a}_1$ ,  $\vec{b}_1$ , and  $\vec{c}_1$  surfaces when  $K=2$ . This causes the point distributions on the surface, nose, top, and bottom grids to have the major influence over the distribution of grid points within the interior of the volume grid. A few intermediate grids are shown in Fig. 2.16 to demonstrate the results of the transfinite interpolation.

The grid is constructed with 101 points in the  $\xi$ -direction, 51 points in the  $\eta$ -direction, and 35 points in the  $\zeta$ -direction. This grid size is suitable for Euler calculations which are discussed in the following chapters. Figure 2.17 demonstrates the mapping from the physical domain to the unit cube computational domain.

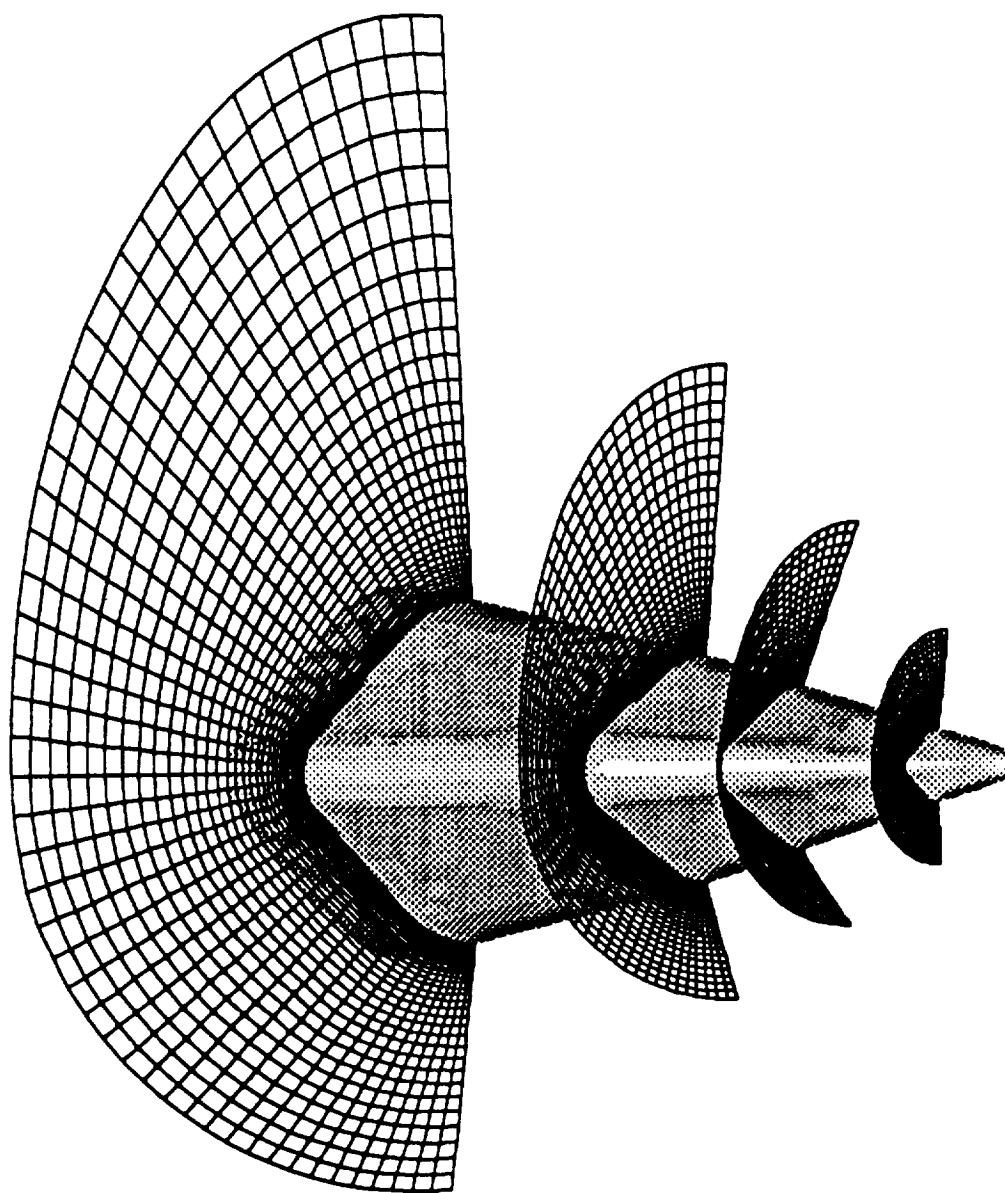


Fig. 2.16 Intermediate surfaces resulting from transfinite interpolation.

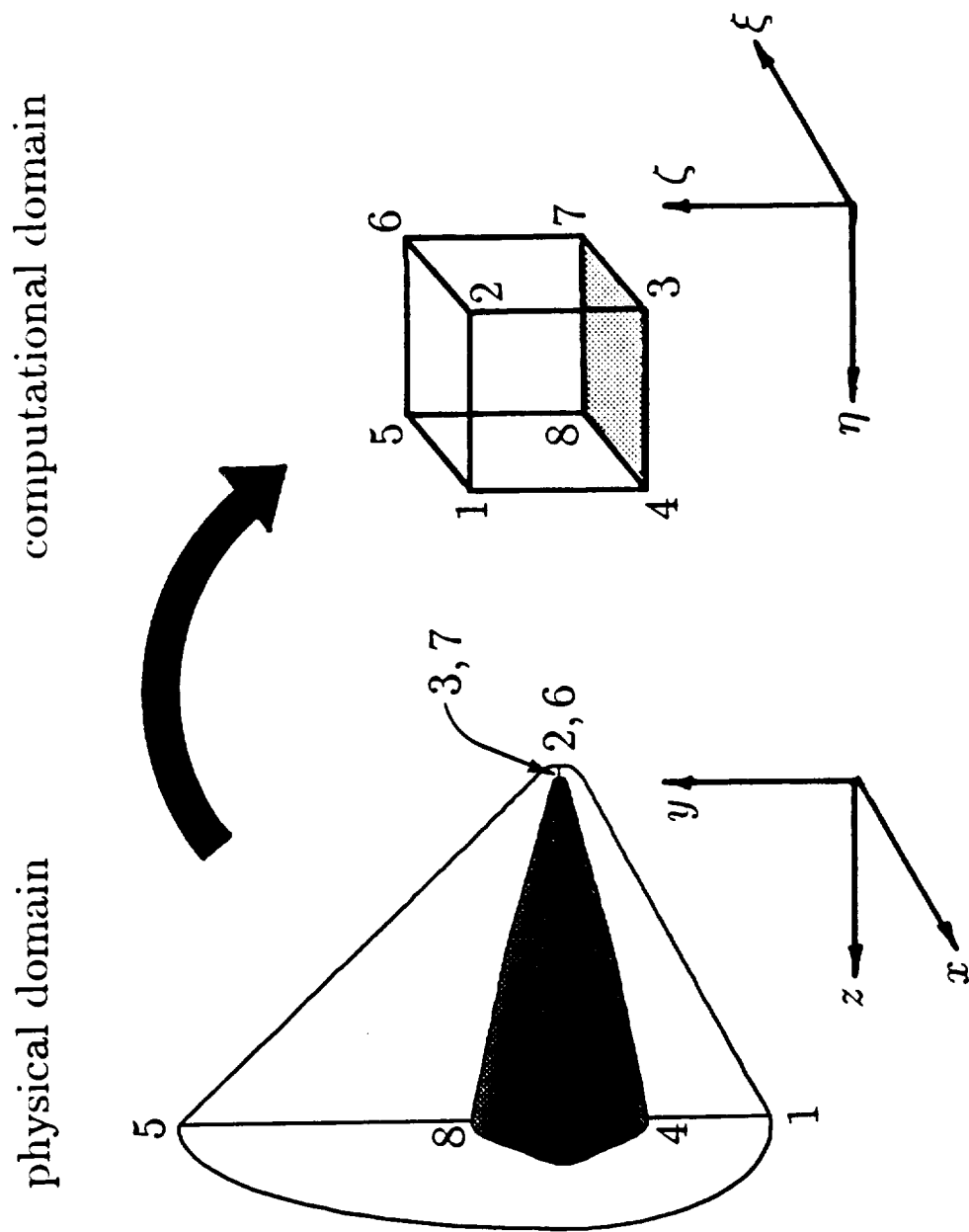


Fig. 2.17 Mapping from physical domain to computational domain.

## Chapter 3

# GOVERNING EQUATIONS AND SOLUTION TECHNIQUE

### 3.1 Introduction

There are a number of methods currently being used to determine the flow characteristics around aircraft and spacecraft. The method used in this study is based on an upwind-biased finite volume algorithm developed by Gnoffo [6,8]. The viscous code based on this algorithm is called the Langley Aerothermodynamic Upwind Relaxation Algorithm (LAURA). This code has been modified by Weilmuenster, Smith and Greene to compute inviscid flowfields [11]. A brief description of the LAURA code and the implementation of the inviscid boundary conditions are given here.

### 3.2 Governing Equations

The governing equations for this code are the three-dimensional Euler equations for a compressible perfect gas. The integral form of these equations is [11]

$$\int \int \int \mathbf{q}_t \, d\Omega + \int \int \mathbf{h} \, d\sigma = 0 \quad . \quad (3.1)$$

Expressing Eq. (3.1) in finite-volume form for a single six-sided cell (Fig. 3.1) in the computational domain gives

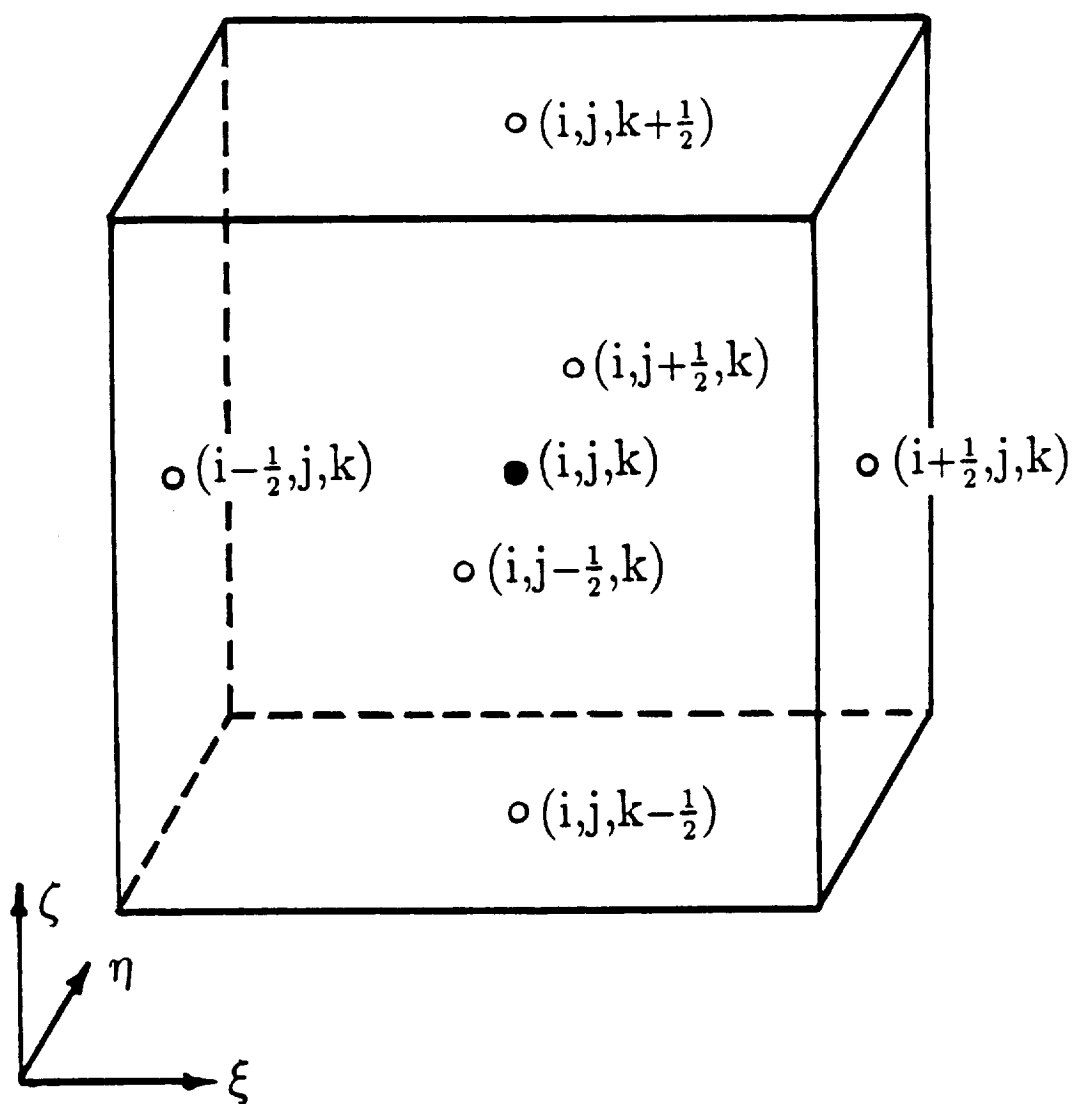


Fig. 3.1 Computational cell.

$$\delta \mathbf{q}_{i,j,k} = - \left[ \frac{\delta t}{\Omega} \right]_{i,j,k} \{ [(\mathbf{E}\sigma)_{i+\frac{1}{2},j,k} - (\mathbf{E}\sigma)_{i-\frac{1}{2},j,k}] + [(\mathbf{F}\sigma)_{i,j,k+\frac{1}{2}} - (\mathbf{F}\sigma)_{i,j,k-\frac{1}{2}}] + [(\mathbf{G}\sigma)_{i,j,k+\frac{1}{2}} - (\mathbf{G}\sigma)_{i,j,k-\frac{1}{2}}] \} \quad (3.2)$$

where  $\delta \mathbf{q}_{i,j,k} = \mathbf{q}_{i,j,k}^{n+1} - \mathbf{q}_{i,j,k}^n$  ;  $\delta t = t_{n+1} - t_n$  .

The variable  $\Omega$  is defined here as the cell volume while  $\sigma$  is the cell face area. The vector  $\mathbf{q}$  is defined as

$$\mathbf{q} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho E_t \end{bmatrix} \quad (3.3)$$

where

$$E_t = e + 1/2(u^2 + v^2 + w^2) \quad (3.4)$$

The variables  $\rho$  and  $e$  in Eqs. (3.3) and (3.4) are the nondimensional values of density and internal energy per unit mass, respectively. The variables  $u$ ,  $v$ , and  $w$  are the Cartesian components of velocity in nondimensional form. The subscripted lower-case letters indicate cell center values, unless offset by a half, in which case they indicate values at a cell face (Fig. 3.1).

The normal inviscid fluxes at a cell face ( $\mathbf{E}$ ,  $\mathbf{F}$ ,  $\mathbf{G}$ ) all have the form of  $\mathbf{H}$  shown below

$$\mathbf{H}_{l+\frac{1}{2}} = \frac{1}{2} [a \mathbf{I}(\mathbf{q}_l)^* + b \mathbf{I}(\mathbf{q}_{l+1})^* - \frac{1}{\nu_{l+\frac{1}{2}}} \mathbf{M}_{l+\frac{1}{2}} \lambda_{l+\frac{1}{2}} (\mathbf{s}_{l+\frac{1}{2}}^* - \theta \mathbf{s}_{l+\frac{1}{2}}^{\min})] \quad (3.5)$$

The asterisk represents the time level and may be either the  $n$  or  $n+1$  level, depending upon the cell face being evaluated. If variables with this superscript happen to contain information referenced in the  $i, j, k$  cell center, these values are linearized using

$$\mathbf{K}_{i,j,k}^{n+1} = \left[ \mathbf{K}^n + \frac{\partial \mathbf{K}}{\partial \mathbf{q}} \right] \delta \mathbf{q}_{i,j,k} \quad (3.6)$$

where  $\mathbf{K}$  is a dummy variable representing the value to be linearized. If these linearizations are not included, an explicit rather than an implicit algorithm is created. The functions  $a_w$  and  $b_w$  are weighting parameters defined in terms of cell volumes as

$$\begin{aligned} a_w &= 2 \frac{\Omega_{l+1}}{\Omega_l + \Omega_{l+1}} , \\ b_w &= 2 \frac{\Omega_l}{\Omega_l + \Omega_{l+1}} . \end{aligned} \quad (3.7)$$

The inclusion of the  $a_w$  and  $b_w$  parameters lessens the effects of grid stretching near the axis singularity and in other regions where the grid is highly stretched [7]. The choice of  $a_w$  and  $b_w$  is empirical, and other formulations are suggested in Ref. 6. For any cell face, the inviscid normal flux,  $\mathbf{I}$ , is computed using Roe's averages of cell-centered values and has the form

$$\mathbf{I} = \begin{bmatrix} \rho U \\ \rho u U + P n_x \\ \rho v U + P n_y \\ \rho w U + P n_z \\ (\rho E_t + P) U \end{bmatrix} . \quad (3.8)$$

In this equation,  $P$  is the nondimensional pressure and  $U$  is the contravariant velocity normal to the cell face. Variables such as the unit vector normal to a cell face,  $\mathbf{n} = n_x \hat{\mathbf{i}} + n_y \hat{\mathbf{j}} + n_z \hat{\mathbf{k}}$ , inverse distance between cell centers,  $\nu$ , eigenvalue matrix,  $\lambda$ , and the right and left eigenvector matrices,  $\mathbf{M}^{-l}$  and  $\mathbf{M}$ , can be found in Ref. 6. This reference also contains the definitions of the unit vectors tangent to a cell face, the cell volume, face area, and timestep.

A first-order-accurate flux is computed when  $\theta$  is equal to zero in Eq. (3.5), while a second-order-accurate flux is computed when  $\theta$  is equal to one. Because the two terms are explained in detail in Ref. 6, only a brief outline is given here. The flux shown in Eq. (3.5) can be thought of as a second-order approximation of the flux at a cell face (first two terms), minus a dissipation term (remaining terms). If this dissipation is not included, the algorithm is equivalent to a centrally-differenced

algorithm. The first-order dissipation is given as

$$\frac{1}{\nu} \mathbf{M} |\lambda| \mathbf{s}^* \quad , \quad (3.9)$$

where

$$\mathbf{s}^* = \mathbf{M}^{-l} \Delta \mathbf{q}^* \quad .$$

The change in  $\mathbf{q}$  across a cell face,  $\Delta \mathbf{q}$ , is computed via the upwind differencing method attributed to Roe [17]. The variable  $\nu$  is included here for the same reason  $a$  and  $b$  are included in Eq. (3.5). The matrix  $\lambda$  contains the absolute values of the eigenvalues. Roe's first-order dissipation term is the exact solution to the approximate initial value Riemann problem which is

$$\mathbf{q}_t + \mathbf{J} \Delta \mathbf{q} = 0 \quad . \quad (3.10)$$

These three matrices ( $\mathbf{M}$ ,  $\mathbf{M}^{-l}$ , and  $\lambda$ ) relate to the inviscid flux vector Jacobian  $\mathbf{J}$  by the equation

$$\mathbf{J} = \frac{\partial \mathbf{I}}{\partial \mathbf{q}} = \mathbf{M} \lambda \mathbf{M}^{-l} \quad . \quad (3.11)$$

Roe's first-order dissipation term can be thought of as two flux differences across a cell face – one flux difference associated with the positive eigenvalues and the other associated with the negative eigenvalues, such that

$$\mathbf{M} |\lambda| \mathbf{M}^{-l} \Delta \mathbf{q}^* = \Delta \mathbf{I}^+ - \Delta \mathbf{I}^- = |\mathbf{J}| \Delta \mathbf{q}^* \quad . \quad (3.12)$$

Equation (3.12) is an exact equality if the matrices are evaluated using the Roe's averaged variables. For a nondimensional case, Roe's method interprets the inviscid zone of dependence correctly. Although not provided, the same is assumed for a three-dimensional flow. Allowing this assumption, if a flow is supersonic and has all positive eigenvalues, a flux is constructed based entirely on the upstream information. Problems with eigenvalues of mixed signs are handled accordingly.

Second-order accuracy is achieved with the Symmetric Total Variation Diminishing (STVD) scheme of Yee [18]. In this scheme, gradients of characteristic variables are compared and selected such that no extraneous maxima or minima are introduced. This is accomplished through the use of a minmod function. This function compares two values and returns the smallest in absolute magnitude if the signs are the same, or returns zero if the signs are different. The minmod portion of the second-order term in Eq. (3.5) is

$$\mathbf{s}^{min} = \text{minmod}(\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3) \quad . \quad (3.13)$$

The subscript 2 references the face at which the flux is being computed while 1 is the face behind and 3 is the face ahead.

The implicit algorithm is therefore written as

$$\begin{aligned} & \left\{ \mathbf{I} + \frac{\delta t}{2\Omega} \left[ (|\mathbf{A}|_{i-\frac{1}{2},j,k} - \mathbf{A}_{i,j,k}) \sigma_{i-\frac{1}{2},j,k} + (|\mathbf{A}|_{i+\frac{1}{2},j,k} + \mathbf{A}_{i,j,k}) \sigma_{i+\frac{1}{2},j,k} \right. \right. \\ & \quad + (|\mathbf{B}|_{i,j-\frac{1}{2},k} - \mathbf{B}_{i,j,k}) \sigma_{i,j-\frac{1}{2},k} + (|\mathbf{B}|_{i,j+\frac{1}{2},k} + \mathbf{B}_{i,j,k}) \sigma_{i,j+\frac{1}{2},k} \\ & \quad \left. \left. + (|\mathbf{C}|_{i,j,k-\frac{1}{2}} - \mathbf{C}_{i,j,k}) \sigma_{i,j,k-\frac{1}{2}} + (|\mathbf{C}|_{i,j,k+\frac{1}{2}} + \mathbf{C}_{i,j,k}) \sigma_{i,j,k+\frac{1}{2}} \right] \right\} \delta \mathbf{q}_{i,j,k} \\ & = \frac{\delta t}{\Omega} \left[ \frac{1}{2} (b \mathbf{e}_{i,j,k}^n + a \mathbf{e}_{i-1,j,k}^* - |\mathbf{A}|_{i-\frac{1}{2},j,k} (\mathbf{q}_{i-1,j,k}^* - \mathbf{q}_{i,j,k}^n)) \sigma_{i-\frac{1}{2},j,k} \right. \\ & \quad - \frac{1}{2} (b \mathbf{e}_{i+1,j,k}^* + a \mathbf{e}_{i,j,k}^n - |\mathbf{A}|_{i+\frac{1}{2},j,k} (\mathbf{q}_{i,j,k}^n - \mathbf{q}_{i+1,j,k}^*)) \sigma_{i+\frac{1}{2},j,k} \\ & \quad + \frac{1}{2} (b \mathbf{f}_{i,j,k}^n + a \mathbf{f}_{i,j-1,k}^* - |\mathbf{B}|_{i,j-\frac{1}{2},k} (\mathbf{q}_{i,j-1,k}^* - \mathbf{q}_{i,j,k}^n)) \sigma_{i,j-\frac{1}{2},k} \\ & \quad - \frac{1}{2} (b \mathbf{f}_{i,j+1,k}^* + a \mathbf{f}_{i,j,k}^n - |\mathbf{B}|_{i,j+\frac{1}{2},k} (\mathbf{q}_{i,j,k}^n - \mathbf{q}_{i,j+1,k}^*)) \sigma_{i,j+\frac{1}{2},k} \\ & \quad + \frac{1}{2} (b \mathbf{g}_{i,j,k}^n + a \mathbf{g}_{i,j,k-1}^* - |\mathbf{C}|_{i,j,k-\frac{1}{2}} (\mathbf{q}_{i,j,k-1}^* - \mathbf{q}_{i,j,k}^n)) \sigma_{i,j,k-\frac{1}{2}} \\ & \quad \left. - \frac{1}{2} (b \mathbf{g}_{i,j,k+1}^* + a \mathbf{g}_{i,j,k}^n - |\mathbf{C}|_{i,j,k+\frac{1}{2}} (\mathbf{q}_{i,j,k}^n - \mathbf{q}_{i,j,k+1}^*)) \sigma_{i,j,k+\frac{1}{2}} \right] . \quad (3.14) \end{aligned}$$

Here,  $\mathbf{e}$ ,  $\mathbf{f}$ , and  $\mathbf{g}$  are the normal fluxes in the  $\xi$ ,  $\eta$ , and  $\zeta$ -coordinate directions, respectively, while  $\mathbf{A}$ ,  $\mathbf{B}$ , and  $\mathbf{C}$  are their corresponding inviscid flux Jacobians. The absolute value Jacobians are computed using the Roe's averaged variables and because  $\mathbf{q}$  is updated in planes parallel to the body, the scheme is point Jacobi within

the plane. This is a direct result of the fact that values used within the plane are all at the same time level. Values located above or below the plane being updated may be either the  $n$  or  $n+1$  time level. These time levels are represented by the asterisk which remains in the formulation. For more details regarding the algorithm and relaxation strategies, one should consult Ref. 6.

### 3.3 Boundary Conditions

Figure 3.2 gives a schematic of a typical wall boundary. In this figure, (a) denotes the plane of cell-centered values a half cell above the body surface, (b) denotes the plane of cell-centered values a half cell below the body surface, and  $x$  denotes the location of points on the surface. The values at (b) are required for the computation of the first and second-order dissipation associated with the Roe and Yee methods, respectively. The values used at (b) are determined by simply equating them with the values used at (a).

Values on the wall are determined such that surface tangency is observed. This method extrapolates values to the wall, and a wave correction is then performed on the values so that surface tangency is satisfied. In general, just extrapolating values to the surface will not meet this requirement. In order to determine the corrected values (c) at the wall, predicted values at the wall are found by first-order extrapolation using the values at (a). These values, along with the wave correction equations [19]

$$\begin{aligned} \frac{P_a}{\rho_a^\gamma} &= \frac{P_c}{\rho_c^\gamma} , \\ U_a - 2\frac{C_a}{\gamma-1} &= U_c - 2\frac{C_c}{\gamma-1} , \\ U_c &= 0 , \\ V_a &= V_c , \end{aligned}$$

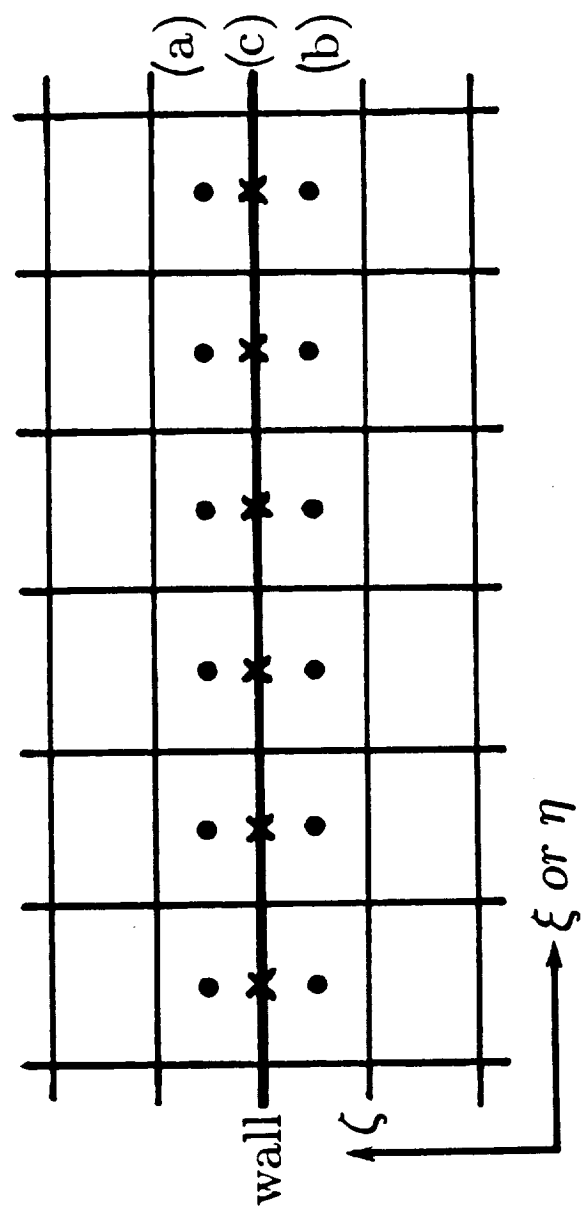


Fig. 3.2 Wall boundary.

$$W_a = W_c \quad , \quad (3.15)$$

are then used to determine the appropriate values at the wall. Here,  $U$  is the contravariant velocity normal to the body surface,  $V$  and  $W$  are the contravariant velocities tangent to the surface, and  $C$  is the speed of sound.

### 3.4 Aerodynamic Quantities

There are a number of different aerodynamic quantities which are used to describe the aerodynamic characteristics of a vehicle. Many of these quantities are based on the surface forces acting on the vehicle. The total surface force vector,  $\vec{F}$  is given in integral form as

$$\vec{F} = \int_{\Sigma} P \vec{n} d\sigma \quad , \quad (3.16)$$

where  $P$  is the pressure on the surface,  $\vec{n}$  is a unit vector normal to the surface,  $\sigma$  is an element of surface area, and  $\Sigma$  is the total surface area. The yaw, lift ( $L$ ), and drag ( $D$ ) are found by taking the  $x$ ,  $y$ , and  $z$  components of  $\vec{F}$ , respectively. The coefficient of lift,  $C_l$ , and the coefficient of drag,  $C_d$ , are defined by

$$C_l = \frac{L/A}{\frac{1}{2}\rho_{\infty} |\vec{V}_{\infty}|^2} \quad , \quad (3.17)$$

$$C_d = \frac{D/A}{\frac{1}{2}\rho_{\infty} |\vec{V}_{\infty}|^2} \quad , \quad (3.18)$$

where  $\vec{V}_{\infty}$  is the free stream velocity vector and  $A$  is the base area of the vehicle. The pitching-moment,  $\vec{M}$ , for the vehicle is given by

$$\vec{M} = \int_{\Sigma} P (\vec{n} \times \vec{m}) d\sigma \quad , \quad (3.19)$$

where  $\vec{m}$  is a vector extending from the vehicle's center of gravity to the center of an element of area on the surface. The coefficient of pitching-moment is given by

$$C_m = \frac{\vec{M}/A}{\frac{1}{2}\rho_{\infty} |\vec{V}_{\infty}|^2 l} \quad , \quad (3.20)$$

where  $l$  is the length of the model. The coefficient of pressure,  $C_p$ , is defined by

$$C_p = \frac{P - P_\infty}{\frac{1}{2}\rho_\infty |\vec{V}_\infty|^2} \quad , \quad (3.21)$$

where  $P_\infty$  is the free-stream pressure.

## Chapter 4

# FLOWFIELD COMPUTATIONS AND DISCUSSION OF RESULTS

### 4.1 Introduction

Flowfield computations are performed on the (101x51x35) single-block volume grid using the LAURA code [6,8,11] described earlier. The Cray-2 computer at NASA Langley Research Center is used to make these computations. The code requires 13 Mwds of memory for a grid of 180285 points. Each iteration requires 3.89 CPU seconds if the Jacobians are only updated every 20th iteration. Therefore, 21.6 CPU microseconds are required per iteration per grid point. Computations are performed at Mach 6.0 and angles of attack of 0, 6, and 12 degrees. Free-stream conditions which existed in the 1968 study [1] are used in the LAURA code in order to make an accurate comparison between experimental and computational results.

The base diameter ( $d$  in Figs. 2.10 and 2.12) of the  $11.3^\circ$  inner cone is assumed to be 4.0 inches. This creates a vehicle 8.0 inches in length with a base area of 14.85 square inches. These values are used as the characteristic length and characteristic area of the model, respectively (as was done in the 1968 study). The ratio of specific heats,  $\gamma$ , is taken to be 1.4 for these computations.

Free-stream velocities of Mach 6.0 fall within the hypersonic range of fluid flow. A great deal of research has been done in the field of hypersonic flow over the

last few years. Blunt-nosed vehicles in particular have shown a number of common aerodynamic traits [12]. One of these is the formation of a strong normal shock in front of the vehicle. Immediately behind this shock is a region of subsonic flow with hypersonic and supersonic flow in the surrounding regions (Fig. 4.1). This type of flow is also characterized by a highly compressed region in front of the nose, overexpansion around the nose, and recompression downstream of the nose. Hypersonic flow about a cone is also well documented [12]. It is characterized by the formation of a strong shock wave coming off the cone with supersonic and hypersonic flow behind the shock.

Because this study makes use of the Euler equations to determine the flow characteristics around the vehicle, diffusion and thermal conductivity effects are not accounted for. Values of drag and pitching moment should be below those determined experimentally since only form drag, and not viscous drag, is computed by the Euler equations. Also, the change in enthalpy normal to the surface is assumed to be constant. Energy interaction by means of chemical reaction, radiation, molecular rotation, and molecular vibration are not accounted for within the flowfield. Therefore, temperatures on the surface found computationally are expected to be higher than those found experimentally.

## 4.2 Cases Studied

Euler computations are performed on the vehicle at three angles of attack ( $\alpha = 0, 6$ , and  $12$  degrees) and assuming a free-stream Mach number of  $6.0$ .

### Case 1: $0^\circ$ angle of attack

The contour plots of the coefficient of pressure ( $C_p$ ) on the surface, symmetry planes, and exit plane are given in Fig. 4.2 for  $\alpha = 0^\circ$ . A strong normal shock wave forms just in front of the vehicle. The normal shock becomes an oblique shock and continues to the exit plane where the conical shape of the shock becomes apparent.

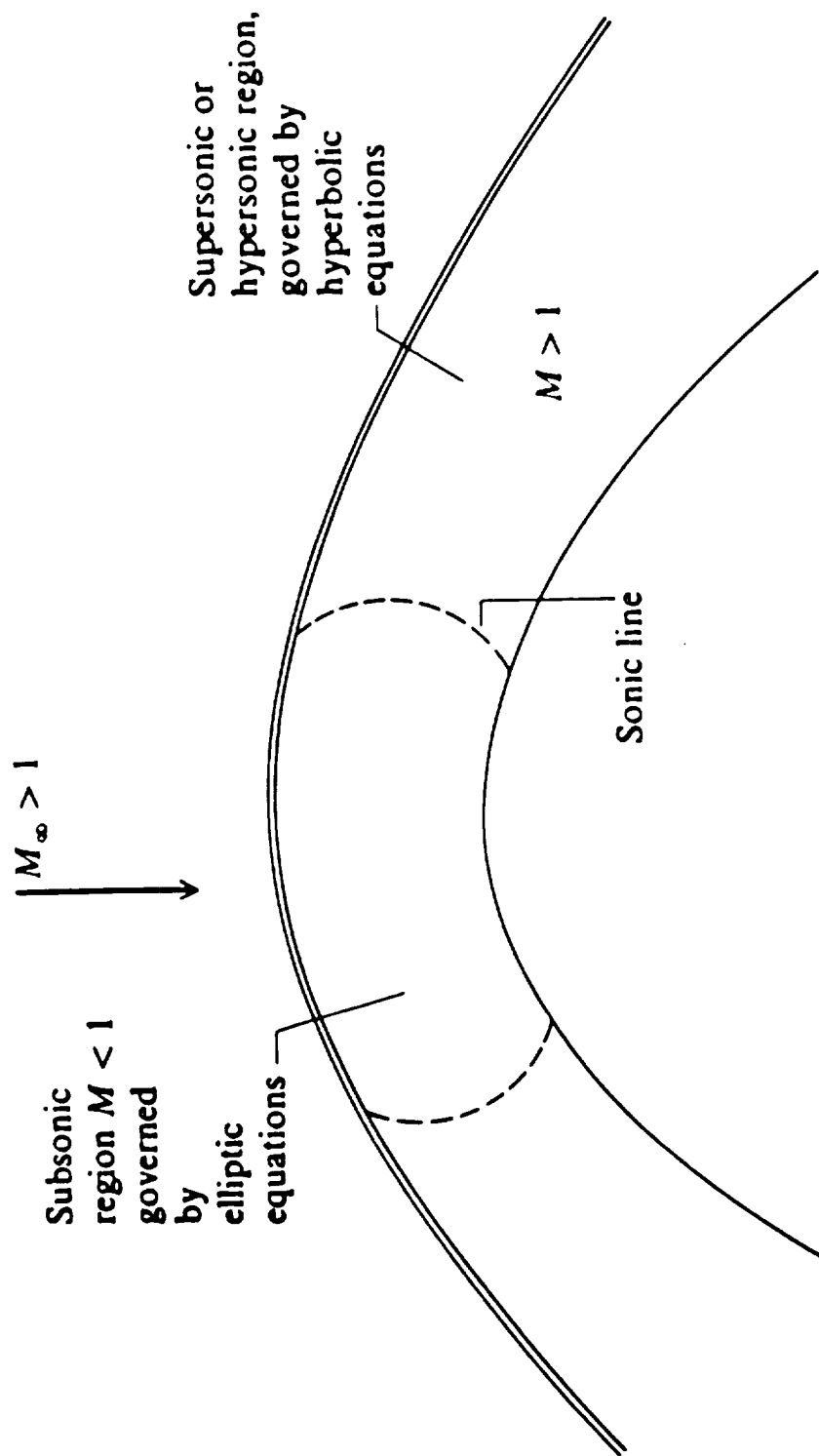


Fig. 4.1 Hypersonic flow around a blunt body.

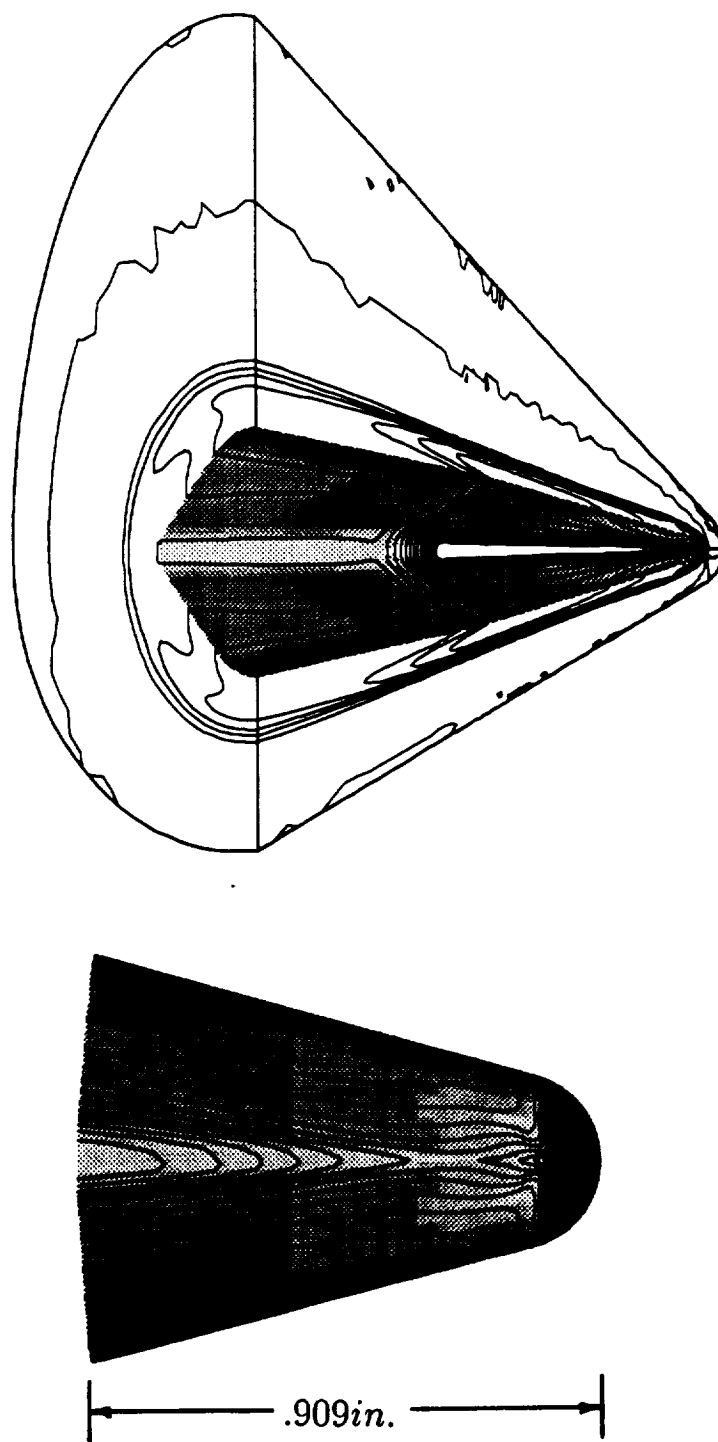


Fig. 4.2  $C_p$  contours for Mach 6.0,  $\alpha = 0^\circ$ .

The pressure contours throughout the flowfield are symmetrical, which is expected for an angle of attack of  $0^\circ$ . It is interesting to note a region of high pressure surrounded by regions of lower pressure on the outer cones of the forebody. An enlarged view of the  $C_p$  contours on the first thirty  $\eta$  lines of the surface is also given in Fig. 4.2. The black region of the nose indicates the very high pressure gradient which is expected in this region. Also demonstrated in this figure are the low pressure regions slightly downstream of the nose. They result from the overexpansion out of the high pressure region in front of the nose. The coefficients of lift, drag, and pitching moment found computationally are  $-2.776 \times 10^{-5}$ , 0.1373, and  $4.128 \times 10^{-5}$ , respectively, while these values found experimentally are 0.0, 0.145, and  $-5.0 \times 10^{-4}$ , respectively. This computed lift is near 0.0 computationally which is expected for  $0^\circ$  angle of attack. The magnitude of the coefficients of drag and pitching moment, however, are underestimated because viscous effects are not accounted for.

#### Case 2: $6^\circ$ angle of attack

A plot of  $C_p$  contours is given in Fig. 4.3 for  $\alpha = 6^\circ$ . This figure shows a strong shock on the windward side of the vehicle. A close examination of Fig. 4.3 compared to Fig. 4.2 shows that the windward shock has moved slightly closer to the body while the leeward shock has moved slightly away. An enlarged view of the nose of the vehicle shows closed pressure contours on the windward side. These contours are again the result of overexpansion near the nose. The coefficients of lift, drag, and pitching moment found computationally are 0.2455, 0.1659, and  $-1.001 \times 10^{-3}$ , respectively, while these values found experimentally are 0.245, 0.18, and  $-2.5 \times 10^{-3}$ , respectively. The coefficient of lift compares very well with the experimental values but the coefficient of drag and pitching moment are below those found experimentally, as expected.

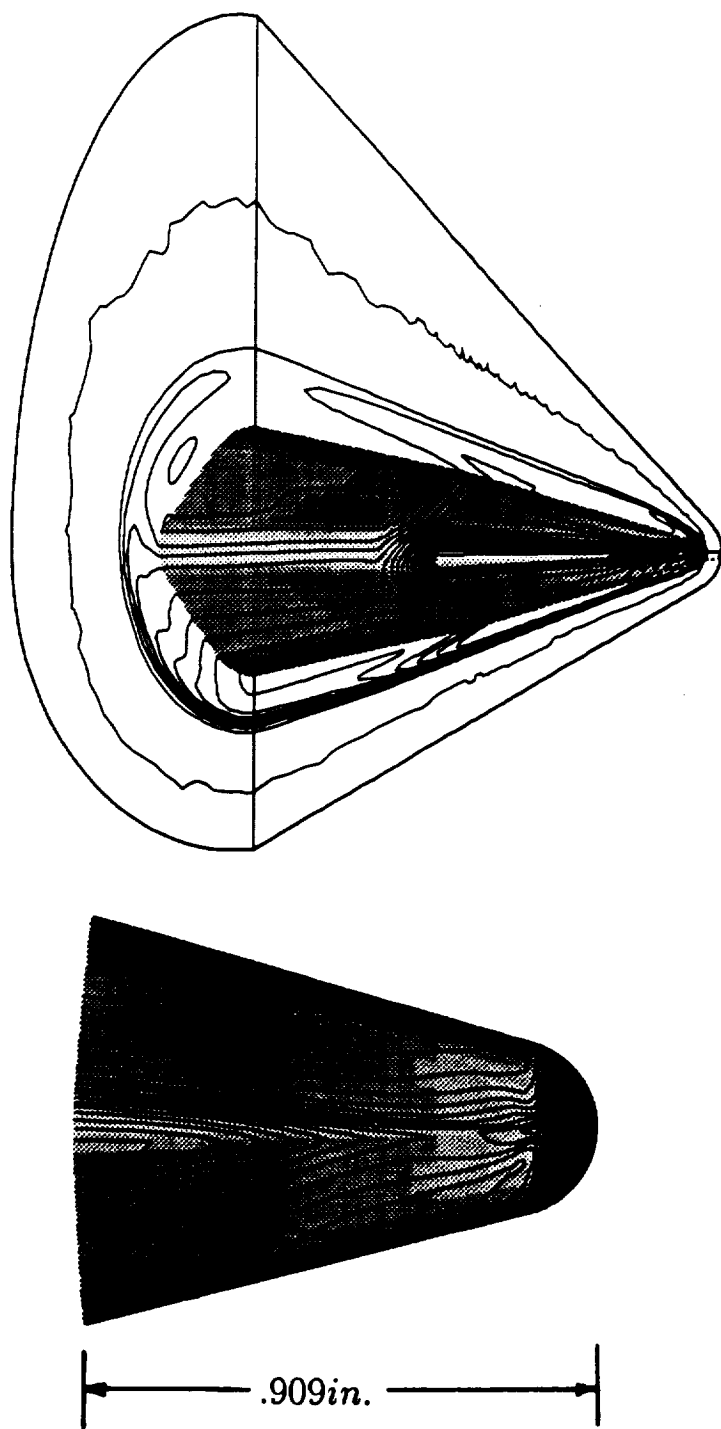


Fig. 4.3  $C_p$  contours for Mach 6.0,  $\alpha = 6^\circ$ .

### Case 3: 12° angle of attack

A plot of  $C_p$  contours is given in Fig. 4.4 for  $\alpha = 12^\circ$ . A strong shock has formed on the windward side of the vehicle. When compared to the  $\alpha = 6^\circ$  case, the windward shock has moved closer to the vehicle while the leeward shock has moved farther away from the vehicle. There are also large pressure gradients located on the outer cones of the forebody. These result from the fact that the outer cones are becoming leading edges at higher angles of attack. The forebody/afterbody junction seems to cause large pressure gradients at higher angles of attack as well. The enlarged view of the model shows the closed contours of low pressure near the nose which are caused by overexpansion in this region. The low pressure has continued to move towards the nose on the windward side as the angle of attack has increased. The coefficients of lift, drag, and pitching moment found computationally are 0.4755, 0.2727, and  $-4.739 \times 10^{-3}$ , respectively, while these values found experimentally are 0.482, 0.30, and  $-5.0 \times 10^{-3}$ , respectively. The coefficient of lift for this case and the two other cases, compares very well with the experimental results. The coefficient of drag and pitching moment, however, are again below those found experimentally.

## 4.3 Comparisons

Figures 4.5 through 4.8 illustrate the computational and experimental values of the coefficients of lift, drag, and pitching moment, as well as lift-over-drag for the three cases studied. Figure 4.5 in particular shows that values of the coefficient of lift found computationally compare very well with those found experimentally. Figure 4.6, however, shows that the magnitudes of the coefficient of drag found computationally are less than those found experimentally. This is due to the fact that viscous effects are not taken into account. The magnitudes of the coefficient of pitching-moment about the x-axis found computationally are also less than the values found experimentally. This is again the result of viscous effects not being taken into account. Corrected

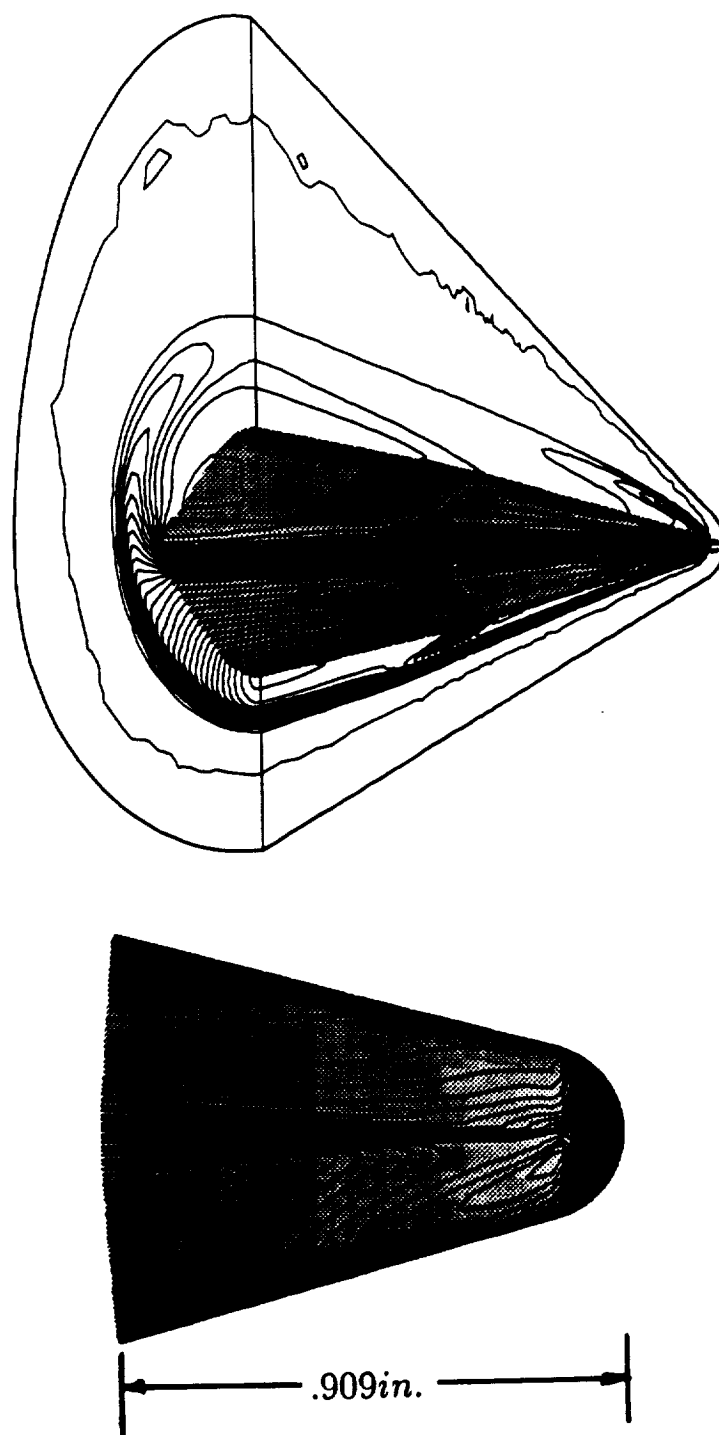


Fig. 4.4  $C_p$  contours for Mach 6.0,  $\alpha = 12^\circ$ .

values for the coefficients of drag and pitching-moment are plotted in Figs. 4.6 and 4.7, respectively. They are found by adding the difference between the computational and experimental values at  $0^\circ$  angle of attack to the remaining computational values. This gives a slight correction to allow for viscous effects. Figure 4.8 compares the computational lift-over-drag ratios to those found experimentally. Since drag is underestimated computationally, lift-over-drag is overestimated. The lift-over-drag at  $0^\circ$  angle of attack, however, is accurate since the value of lift is virtually zero.

Figure 4.9 gives a plot of the density along the stagnation line of the vehicle at  $0^\circ$  angle of attack. This figure shows a strong normal shock coming off the vehicle. The standoff distance of this shock is approximately 0.03 inches. Figures 4.10, 4.11, and 4.12 give the distribution of  $C_p$  along the top, bottom, and side of the model, respectively. Figure 4.10 shows that as the angle of attack increases, the pressure on the top of the vehicle (which is on the leeward side at positive angles of attack) decreases. At  $0^\circ$  angle of attack, a slight dip in the pressure distribution is seen near the nose. This is a result of the overexpansion which occurs in this region. A reduction in pressure is also seen at 4.0 inches which is the location of the forebody/afterbody junction. This is the result of an expansion which occurs in this region. Figure 4.11 gives a plot of  $C_p$  along the bottom of the vehicle. This figure illustrates that as the angle of attack increases, the pressure increases. The overexpansion near the nose becomes greater on the bottom of the vehicle at higher angles of attack. The expansion at the forebody/afterbody junction is also greater at higher angles of attack. Figure 4.12 shows the variation of  $C_p$  along the side of the vehicle for  $0^\circ$  angle of attack. The overexpansion and recompression near the nose is clearly seen in this figure. Also, since there is a greater discontinuity in slope on the side of the model, a greater expansion occurs there. This expansion is shown by the large reduction in pressure at this point.

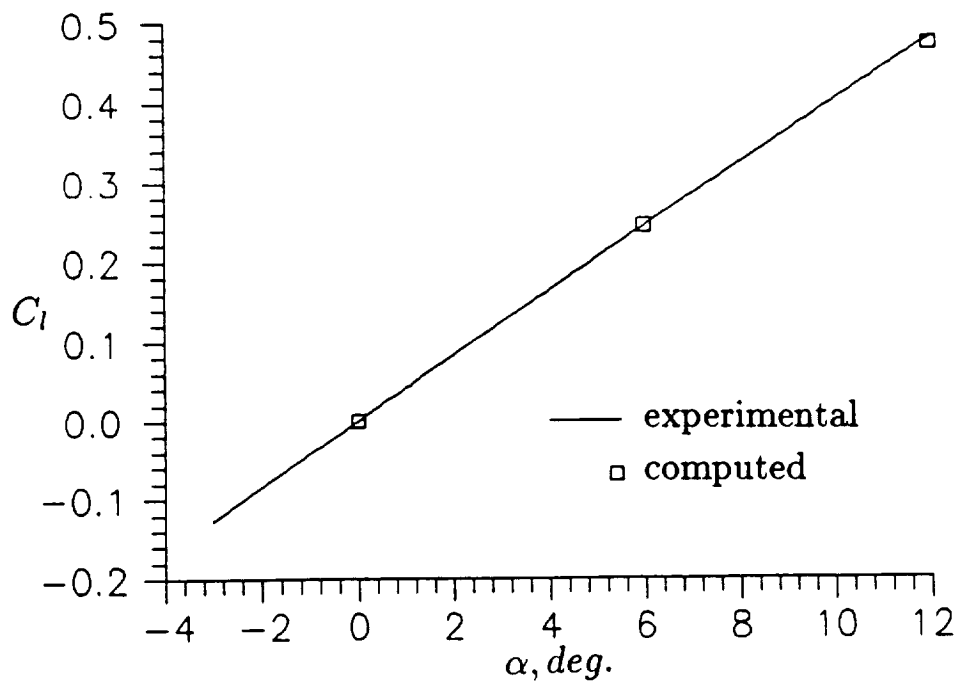


Fig. 4.5 Coefficient of lift vs. angle of attack, Mach 6.0.

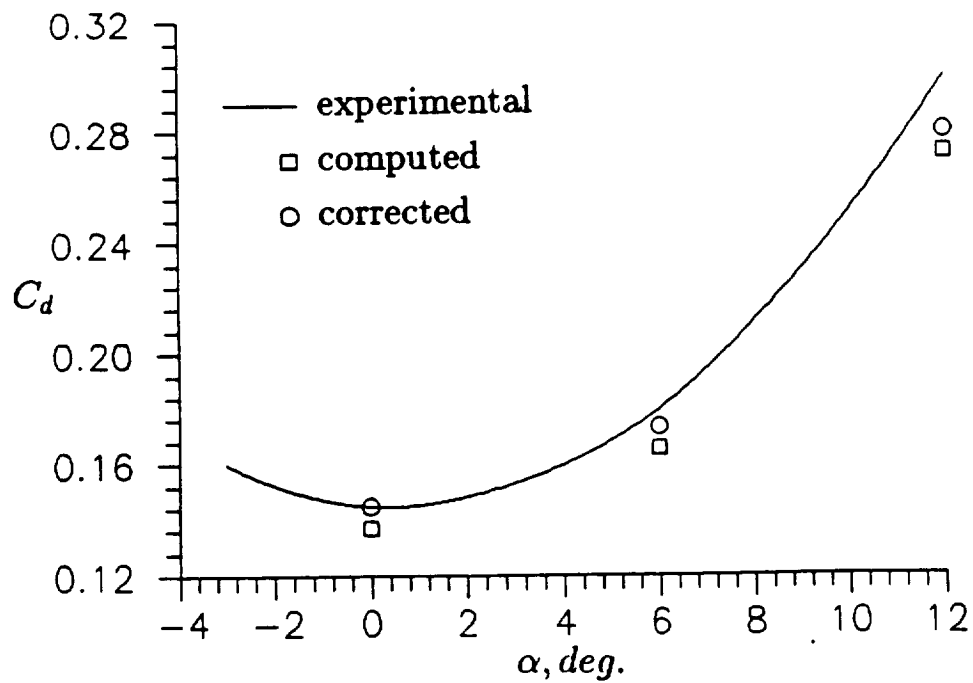


Fig. 4.6 Coefficient of drag vs. angle of attack, Mach 6.0.

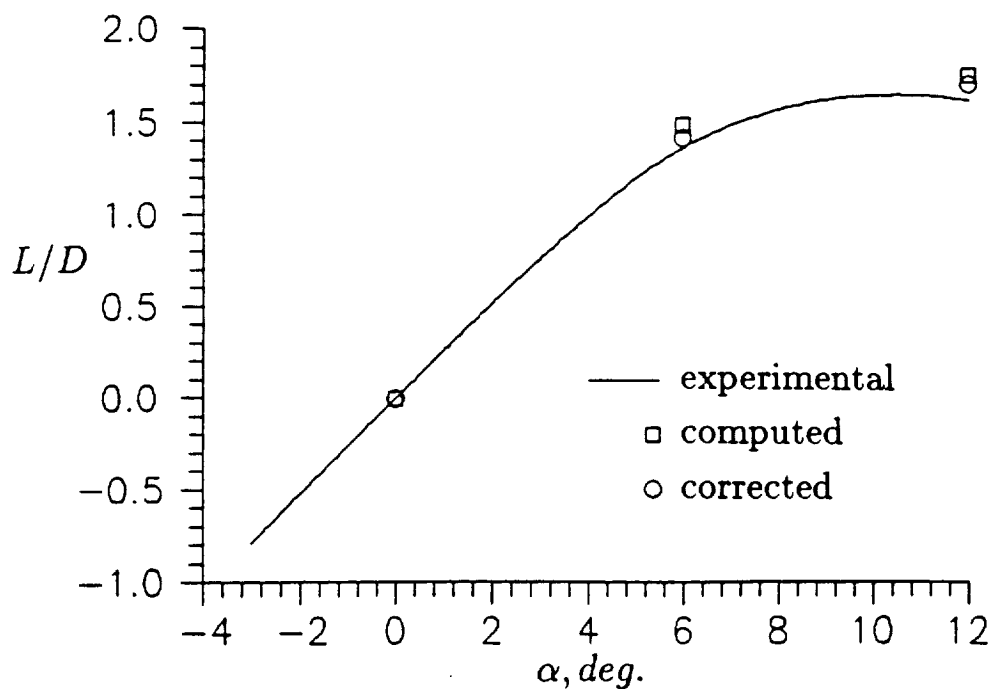


Fig. 4.7 Lift-over-drag vs. angle of attack, Mach 6.0.

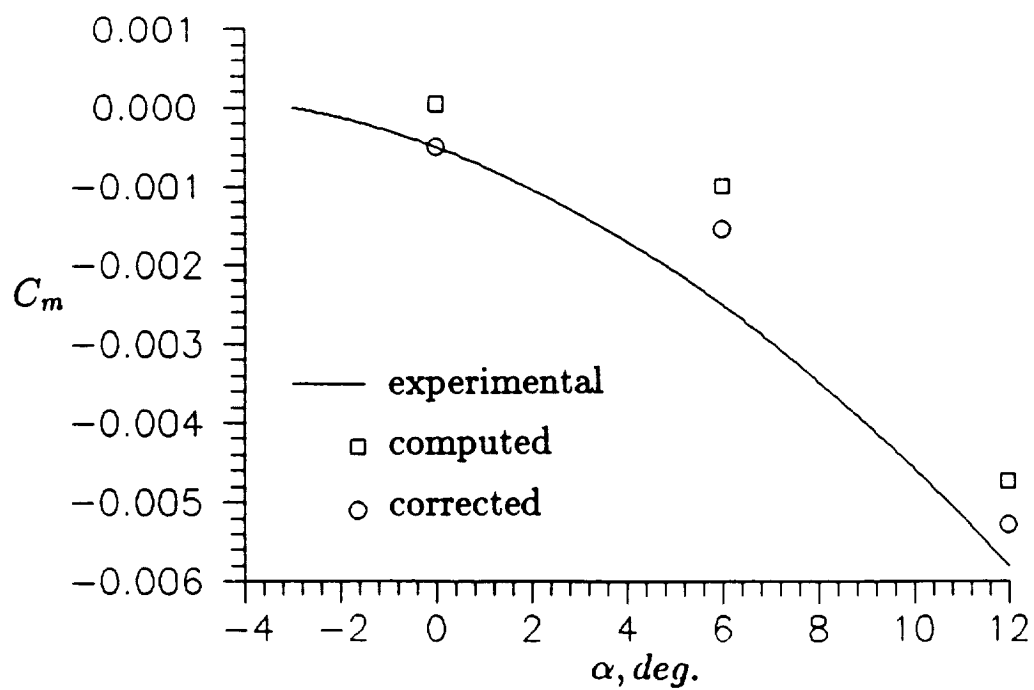


Fig. 4.8 Coefficient of pitching-moment vs. angle of attack, Mach 6.0.

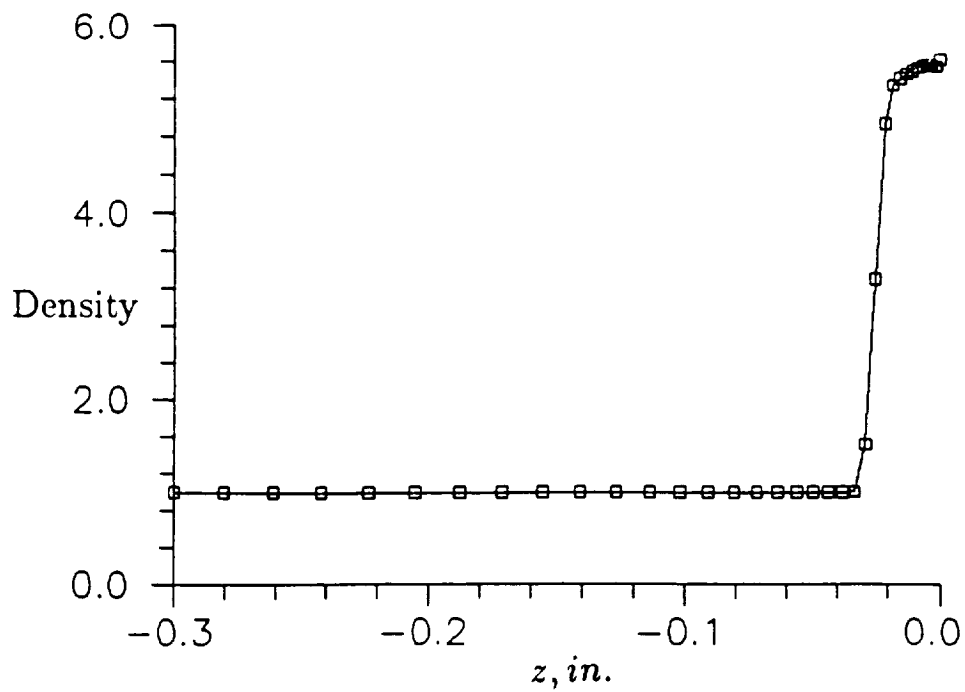


Fig. 4.9 Density vs. distance along stagnation line, Mach 6.0,  $\alpha = 0^\circ$ .

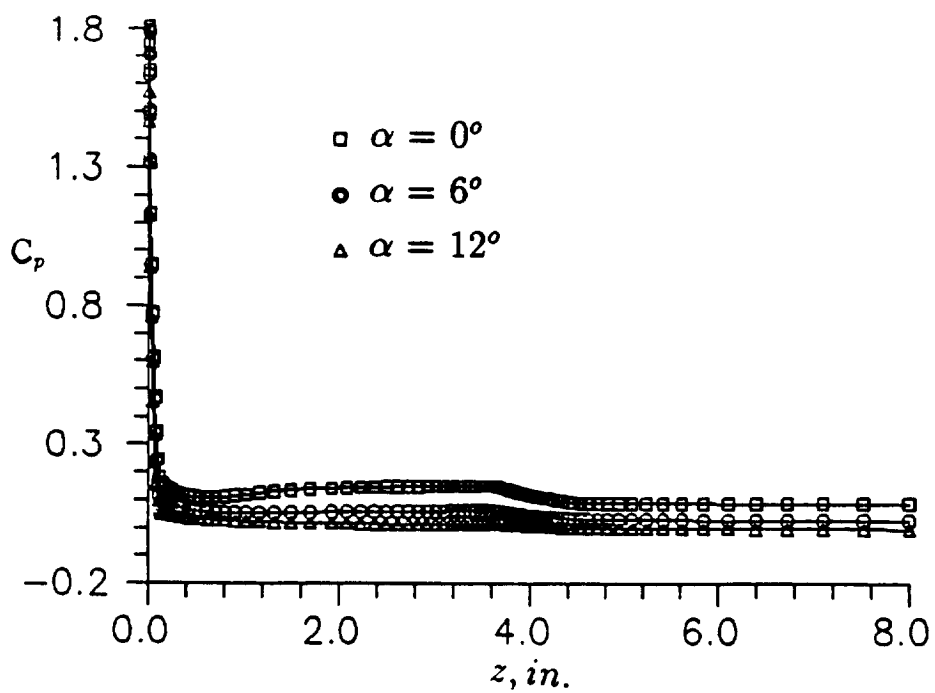


Fig. 4.10  $C_p$  vs. distance along  $z$ -axis, top of vehicle, Mach 6.0.

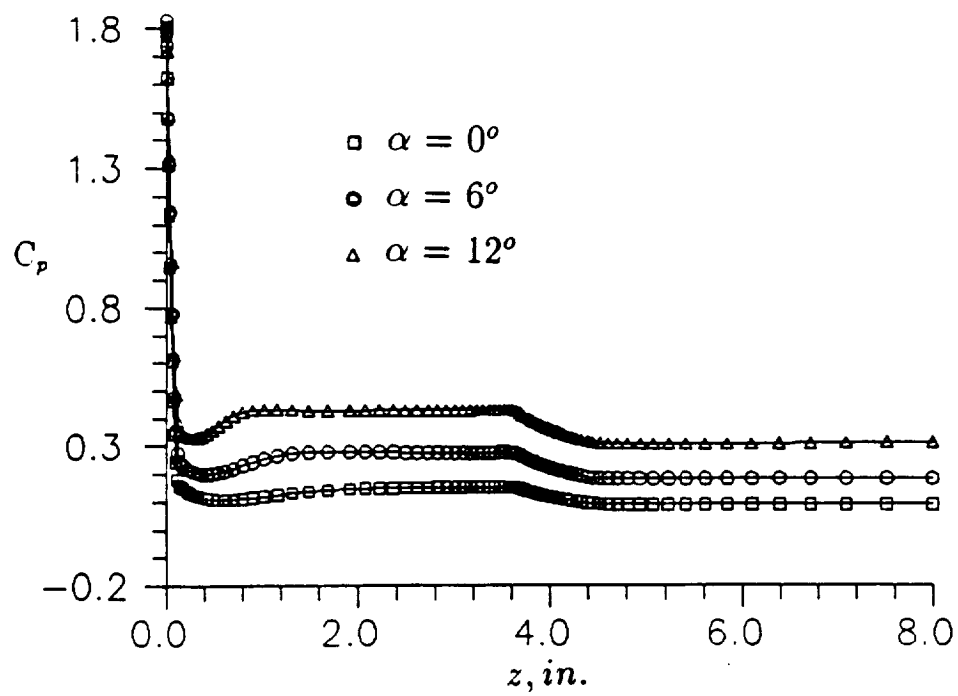


Fig. 4.11  $C_p$  vs. distance along  $z$ -axis, bottom of vehicle, Mach 6.0.

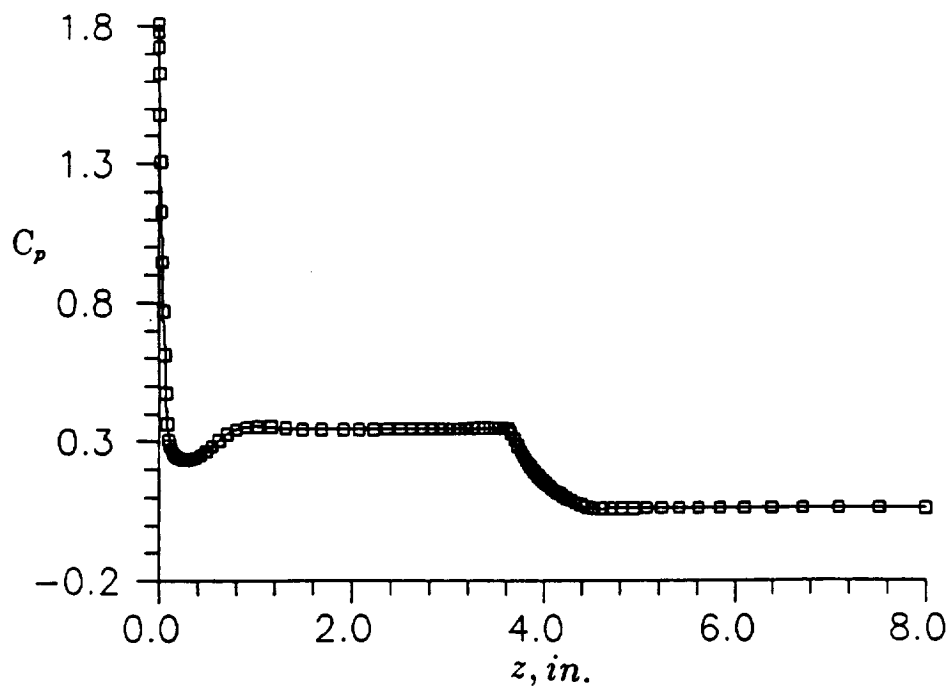


Fig. 4.12  $C_p$  vs. distance along  $z$ -axis, side of vehicle, Mach 6.0,  $\alpha = 0^\circ$ .

## Chapter 5

# Conclusion

The purpose of this study has been to computationally verify the results obtained in 1968 and to continue the study of this vehicle. This study has also, however, helped to further verify the LAURA code. The results obtained agree very well with the experimental results. The lift calculated by the LAURA code agrees almost entirely with the lift computed experimentally. The drag and pitching moment are slightly below the experimental results but they follow the expected trends. The vehicle has a maximum lift-over-drag ratio of 1.75 computationally as compared to a value of 1.57 found experimentally. The pressure distribution on the surface shows that there is a highly compressed region in front of the nose, overexpansion around the nose, and recompression downstream of the nose. An expansion also occurs at the forebody/afterbody junction where there is a change in slope. This expansion is pronounced on the side of the vehicle because of the greater change in slope.

There are currently plans for the continuation of the research on this vehicle. The effects of different angles of attack and Mach numbers will be examined. Also, new vehicles will be constructed with higher degrees of nose bluntness. These vehicles will be tested using different methods to determine the effects that nose bluntness has on the flow characteristics of the vehicle.

## REFERENCES

1. Ashby Jr., G.C. and Staylor, W.F., "Aerodynamic Characteristics of a Modified Cone-Conical-Frustum Entry Configuration at Mach 6.0," NASA TN D-4598, 1968.
2. Thompson, J.F., Warsi, Z.U.A., and Mastin, C.W., Numerical Grid Generation, North-Holland, 1985.
3. Hoffman, Klaus A., Computational Fluid Dynamics For Engineers, Engineering Education System, 1989.
4. Smith, R.E. and Weise, M.R., "Interactive Algebraic Grid-Generation Technique," NASA TP-2533, March 1986.
5. Anderson, D.A., Tannehill, J.C., and Pletcher, R.H., Computational Fluid Mechanics and Heat Transfer, Hemisphere Publishing Corporation, 1984.
6. Gnoffo, P.A., "An Upwind-Biased, Point-Implicit Relaxation Algorithm for Viscous Compressible Perfect-Gas Flow," NASA TP-2953, October 1989.
7. Gnoffo, P.A., McCandless, R.S., and Yee, H.C., "Enhancements to Program LAURA for Computation of Three-Dimensional Hypersonic Flow," AIAA Paper 87-0280, January 1987.
8. Gnoffo, P.A., "Upwind-Biased, Point-Implicit Relaxation Strategies for Viscous, Hypersonic Flows," AIAA Paper 89-1972, June 1989.
9. Weilmuenster, K.J. and Hamilton, H.H. II, "A Comparison of Computed and Measured Aerodynamic Characteristics of a Proposed Aeroassist Flight Experiment Configuration," AIAA Paper 86-1366, June 1986.
10. Gnoffo, P.A. and Greene, F.A., "A Computational Study of the Flowfield Surrounding the Aeroassist Flight Experiment Vehicle," AIAA Paper 87-0280, January 1987.
11. Weilmuenster, R.A., Smith, R.A., and Greene, F.A., "Assured Crew Return Vehicle Flowfield and Aerodynamic Characteristics," AIAA Paper 90-0229, January 1990.
12. Anderson Jr., J.D., Hypersonic and High Temperature Gas Dynamics, McGraw-Hill Book Company, 1989.
13. Reinsch, C.H., "Smoothing by Spline Functions," Numerische Mathematik, Vol. 10, No. 3, February 1967, pp. 177-183.

14. Liepmann, H.W. and Roshko, A., Elements of Gasdynamics, John Wiley & Sons, Inc., 1957.
15. Zucrow, M.J. and Hoffman, J.D., Gas Dynamics, Vol. 1, John Wiley & Sons, Inc., 1976.
16. Gordon, W.J. and Hall, C.A., "Construction of Curvilinear Coordinate Systems and Applications to Mesh Generation," International Journal for Numerical Methods in Engineering, Vol. 7, July 1973, pp. 461-477.
17. Roe, P.L., "Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes," Journal of Computational Physics, Vol. 43, June 1981, pp. 357-372.
18. Yee, H.C., "On Symmetric and Upwind TVD Schemes," NASA TM-86842, September 1985.
19. Osher, S. and Chakravarthy, S., "Upwind Schemes and Boundary Conditions With Applications to Euler Equations in General Geometries," Journal of Computational Physics, Vol. 50, March 1983, pp. 447-481.

APPENDIX A  
SURFACE GENERATION PROGRAMS: ZFIND, SHIP

```

C      PROGRAM ZFIND
C      *****
C      *      JOHN STEWART      FEBRUARY 10, 1990      *
C      *      PROGRAM WHICH FINDS THE Z-VALUES FOR EACH CROSS-SECTION      *
C      *      TO BE USED IN PROGRAM SHIP.  CALCULATIONS ARE BASED      *
C      *      ON ARCLLENGTH OF INNER CONES AND SPHERICAL NOSE.      *
C      *
C      *      NI = NUMBER OF CROSS-SECTIONS TO BE FOUND (MUST      *
C      *      MATCH NI IN PROGRAM SHIP)      *
C      *      NS = NUMBER OF ARC LENGTH DIVISIONS USED FOR CALC. 'S'      *
C      *      ZI = LENGTH OF AFTERBODY      *
C      *      ZII = LENGTH FROM INNER CONE TANGENTS TO AFTERBODY      *
C      *      ZIII = LENTH OF SPHERICAL NOSE      *
C      *      D = MAXIMUM AFTERBODY INNER CONE DIAMETER      *
C      *      STMAX = MAXIMUM ARC LENGTH      *
C      *****
C      PARAMETER(NI=200,NS=5000)
C      DIMENSION Z1(NS),AL(NS), Z(NI+1), Y(NI+1)
C      REAL KSP
C      OPEN(6,FILE='ZDATA',FORM='FORMATTED')
C      D=4.0
C      ZI=D
C      ZII=.97039*D
C      ZIII=D-ZII
C
C      DEFINE THREE POINTS FOR THREE EXPONENTIAL CURVES TO FIT
C      SMOOTHLY THROUGH. POINTS ARE (TAU*,AA*). DISTRIBUTIONS
C      OF ARC LENGTH WILL BE CHOSEN OFF AA AXIS FOR EQUAL DIVISIONS
C      OF ARC LENGTH ON TAU AXIS. KSP CONTROLS THE AMOUNT OF
C      CURVITURE OF THE CURVES.
C
C      AA1=.02519
C      AA2=.25
C      AA3=.509
C      TAU1=.13
C      TAU2=.35
C      TAU3=.7
C      KSP=.01
C      STMAX = 2.07703*D
C      CALL SPACE(AA1,AA2,AA3,TAU1,TAU2,TAU3,KSP,NI+1,STMAX,Y)
C
C      LOOP FINDS THE VALUES OF ARC LENGTH FOR EACH OF NS
C      CROSS-SECTIONS BASED ON DISTRIBUTION FOUND ABOVE
C
C      DO 31 I=1, NS
C      Z1(I) = 2.0*D*(FLOAT(I-1)/FLOAT(NS-5))
C      IF (Z1(I).LT..02961*D) THEN
C      AL(I) = .04*D*ACOS((.04*D-Z1(I))/(.04*D))
C      GO TO 31
C      ELSE IF (Z1(I).LT.(1.0*D)) THEN
C      AL(I) = .05231*D+1.035616*(Z1(I)-ZIII)
C      GO TO 31
C      ELSE
C      AL(I)=1.05726*D+(Z1(I)-ZII-ZIII)*1.019769
C      GO TO 31
C      ENDIF

```

```

31 CONTINUE
C
C      FINDS CROSS-SECTIONAL Z LOCATION BASED ON ARC LENGTH
C      DISTRIBUTION
C
      K=1
      DO 26 I = 1, NI+1
28 K = K + 1
      IF(Y(I).GT.AL(K)) THEN
        GO TO 28
      ELSE
        Z(I) = ((Y(I)-AL(K-1))*(Z1(K)-Z1(K-1)))/(AL(K)-AL(K-1)) + Z1(K-1)
        K = K - 1
      ENDIF
      IF(I.GT.1) WRITE(*,*) I, Z(I)-Z(I-1),Z(I)
26 CONTINUE
C
C      WRITES OUT RESULTS TO A FILE ZDATA WHERE I=NI+1 IS THE
C      NOSE OF THE SPACECRAFT
C
      DO 1 I = NI+1, 1, -1
      WRITE(6,*) Z(I)
1 CONTINUE
      STOP
      END

      SUBROUTINE SPACE(A1,A2,A3,TAU1,TAU2,TAU3,K2,N,ST,S1)
C *****
C * SUBROUTINE WHICH FITS A SMOOTH CURVE THROUGH THREE POINTS *
C * USING THREE EXPONENTIAL EQUATIONS OF THE FORM: *
C * X = (EXP(KSP*X)-1) / (EXP(KSP)-1) *
C *****
      DIMENSION S1(N)
      REAL K1,K2,K3,K4,K5,NOM,IX
      ICOUNT=0
      K3=-K2
300 CONTINUE
      TEMP=A1*K2*(TAU2-TAU1)/((A2-A1)*TAU1)
      DETAL=1./(1-EXP(-K2))
      DETAR=K3/(EXP(K3)-1.)
      C=TEMP*DETAL
      DF=DETAR-C
      DFDK=(EXP(K3)-1.-K3*EXP(K3))/(EXP(K3)-1)**2
      DK3=-DF/DFDK
      IF(ABS(DK3).LT..00001) GO TO 200
      K3=K3+DK3
      ICOUNT=ICOUNT+1
      IF(ICOUNT.GT.20) GO TO 200
      GO TO 300
200 CONTINUE
      K4=-K3
301 CONTINUE
      TEMP=A2*K3*(TAU3-TAU2)/((A3-A2)*TAU2)
      DETAL=1./(1-EXP(-K3))
      DETAR=K4/(EXP(K4)-1.)
      C=TEMP*DETAL

```

```

      DF=DETAR-C
      DFDK=(EXP(K4)-1.-K4*EXP(K4))/(EXP(K4)-1)**2
      DK4=-DF/DFDK
      IF(ABS(DK4).LT..00001) GO TO 201
      K4=K4+DK4
      GO TO 301
201  CONTINUE
      K5=-K4
302  CONTINUE
      TEMP=A3*K4*(1.-TAU3)/((1.-A3)*TAU3)
      DETAL=1./(1-EXP(-K4))
      DETAR=K5/(EXP(K5)-1.)
      C=TEMP*DETAL
      DF=DETAR-C
      DFDK=(EXP(K5)-1.-K5*EXP(K5))/(EXP(K5)-1.)**2
      DK5=-DF/DFDK
      IF(ABS(DK5).LT..00001) GO TO 202
      K5=K5+DK5
      GO TO 302
202  CONTINUE
1    CONTINUE
      SCALEX=ST
      SCALEF=SCALEX
      DO 10 I=1,N
      ETA=FLOAT(I-1)/FLOAT(N-1)
      IF(ETA.GE.TAU1) GO TO 2
      TERM=K2/TAU1
      NOM=EXP(TERM*ETA)-1.
      DNOM=EXP(K2)-1.
      F=A1*(NOM/DNOM)
      GO TO 3
2    CONTINUE
      IF(ETA.GT.TAU2) GO TO 4
      TERM=K3/(TAU2-TAU1)
      NOM=EXP(TERM*ETA-TERM*TAU1)-1.
      DNOM=EXP(K3)-1.
      F=A1+(A2-A1)*NOM/DNOM
      GO TO 3
4    CONTINUE
      IF(ETA.GT.TAU3) GO TO 5
      TERM=K4/(TAU3-TAU2)
      NOM=EXP(TERM*ETA-TERM*TAU2)-1.
      DNOM=EXP(K4)-1.
      F=A2+(A3-A2)*NOM/DNOM
      GO TO 3
5    CONTINUE
      TERM=K5/(1.-TAU3)
      NOM=EXP(TERM*ETA-TERM*TAU3)-1.
      DNOM=EXP(K5)-1.
      F=A3+(1-A3)*NOM/DNOM
3    CONTINUE
      IX=SCALEX*ETA
      S1(I)=SCALEF*F
      IF (I.EQ.(35+1)/2) WRITE(*,*) '*****',IX,S1(I)
      IF (I.EQ.(37+1)/2) WRITE(*,*) '*****',IX,S1(I)
10  CONTINUE
      RETURN
      END

```

```

C      PROGRAM SHIP
C      *****
C      *      JOHN STEWART                      FEBRUARY 10, 1990      *
C      *      PROGRAM TO ANALYTICALLY DEFINE THE SURFACE GRID        *
C      *      OF THE SPACESHIP FOUND IN NASA TECHNICAL NOTE:          *
C      *      NASA TN D-4598,      JUNE, 1968.                        *
C      *
C      * NOTE:   IN PROGRAM, SHIP IS ORIENTED SUCH THAT THE NOSE IS   *
C      *           AT (0,0,0) AND THE Z-AXES RUNS DOWN THE LENGTH OF   *
C      *           THE SHIP.  THE X-AXIS INTERSECTS THE CYLINDER AND   *
C      *           OUTER CONE CENTERS.  I=1 IS THE BACK OF THE SHIP    *
C      *           AND I=NI+1 IS THE NOSE OF THE SHIP.                *
C      *           OUTPUT DATA IS ORIENTED SUCH THAT X=Z, Y=X, Z=Y AND *
C      *           I=1 IS AT THE NOSE.                                *
C      *
C      * GIVEN DATA:
C      *
C      *      CONANG1 = AFTERBODY CONE ANGLE  (RADIANS)
C      *      CONANG2 = FOREBODY CONE ANGLE  (RADIANS)
C      *      D = MAXIMUM DIAMETER OF AFTERBODY CONE
C      *      ELLD = DISPLACEMENT BETWEEN CONE EDGE AND CYLINDER CENTER
C      *      IN AFTERBODY MEASURED PERPENDICULAR TO CONE WALLS
C      *      ZI= LENGTH OF AFTERBODY
C      *      ZII= LENGTH FROM INNER CONE TANGENTS TO AFTERBODY
C      *      ZIII= LENGTH FROM SPHERE TO INNER CONE TANGENTS
C      *      ZIV= LENGTH OF SPHERICAL NOSE
C      *      RPIPE = RADIUS OF PIPES IN AFTERBODY MEASURED
C      *      PERPENDICULAR TO CONE WALLS
C      *      RDIV = RADIUS OF CONE AT FOREBODY-AFTERBODY CONNECTION
C      *      ELLD1 = ELLIPSE DISPLACEMENT ON FOREBODY AT CONNECTION
C      *      B1 = MAJOR AXIS OF ELLIPSE ON FOREBODY AT CONNECTION
C      *      A1 = MINOR AXIS OF ELLIPSE ON FOREBODY AT CONNECTION
C      *      XO = RADIUS OF NOSE IN XY PLANE AT INNER NOSE TANGENTS
C      *      XO1 = RADIUS OF NOSE IN XY PLANE AT OUTER CONE TANGENTS
C      *      NI = NUMBER OF DIVISIONS DOWN LENGTH OF SHIP
C      *      N = NUMBER OF POINTS OUTLINING SHIP IN XY-PLANE PER QUAD.
C      *      L = NUMBER OF GRID POINTS TO LEAVE OFF THE SURFACE GRID
C      *      AT THE NOSE.
C      *      *****
C      *      PARAMETER(NI=200,N=50)
C      *      PARAMETER(NN=N+1,L=0)
C      *      DIMENSION Z(1+NI-L), A(NI+1-L)
C      *      +,B(NI+1-L), R(NI+1-L)
C      *      +,XN(NI+1-L), CN(NI+1-L), ED(NI+1-L), CG(NI+1-L)
C      *      +,XC(NI+1-L), Y(NI+1-L,NN*2-1)
C      *      +,X(NI-L+1,NN*2-1), XG(NI+1-L)
C      *      +,YN(NI+1-L), DN(NI+1-L), R2(NN)
C      *      +,B5(NI+1-L), ANG(NN)
C      *      REAL M(NI+1-L), KSP, K1, K2
C
C      *      OUTPUT FILE FORMATTED FOR USE WITH PLOT3D ON THE IRIS
C      *      WORKSTATION
C
C      *      OPEN(6,FILE='SHIPO',FORM='FORMATTED')
C
C      *      INPUT FILE GIVING THE CROSS-SECTIONAL Z LOCATIONS

```

```

C          STARTING FROM THE BACK OF THE AFTERBODY AND MOVING FORWARD
C
C          OPEN(7,FILE='ZDATA',FORM='FORMATTED')
C
C          OUTPUT FILE FORMATTED AS X, Y, Z WHERE X IS NOW THE AXIS
C          OF THE SPACESHIP AND N VALUES OF Y AND Z ARE GIVEN FOR
C          EACH VALUE OF X
C
C          OPEN(9,FILE='SHIPOD',FORM='FORMATTED')
C
C          OUTPUT FILE GIVING THE Y-LOCATION OF THE TANGENCY POINTS
C          ON THE INNER CONE AND OUTER CONE(FOREBODY) OR
C          CYLINDER(AFTERBODY).
C
C          OPEN(10,FILE='XNCN',FORM='FORMATTED')
C
C          CONANG1 = .19722
C          CONANG2 = .26302
C          D = 4.0
C          XO = .038624*D
C          XO1 = .036072*D
C          XODIFF=XO-XO1
C          ZI = D
C          ZII = .9704*D
C          ZIII = (.0296-.022723)*D
C          ZIV = .022723*D
C          RPIPE = .12*D
C          RDIV = .30018*D
C          ELLD = .08*D
C          ELLDISP = ELLD/COS(CONANG1)
C          ELLD1 = .07146*D
C          B1 = .13249*D
C          A1 = .1203*D
C
C          READ IN THE CROSS-SECTIONAL AXIAL LOCATIONS BEGINNING
C          FROM THE BACK OF THE SPACESHIP
C
C          DO 75 I = 1, NI+1-L
C          READ(7,*) Z(I)
75 CONTINUE
C          II=0
C
C          CALCULATIONS OF CONE RADIUS (R), MAJOR (B) AND
C          MINOR (A) AXIS OF ELLIPSES CREATED BY PIPES, AND DISTANCE
C          TO CENTER OF ELLIPSE (XC) FOR AFTERBODY. XG AND CG ARE
C          THE POINTS OF TANGENCY GUESSES FOR SUBROUTINE POINTS.
C
76 II=II+1
C          IF (Z(II).LT.ZII+ZIII+ZIV) GO TO 79
C          ED(II) = ELLDISP
C          A(II) = RPIPE
C          B(II) = A(II)/COS(CONANG1)
C          R(II) = D/2. + (Z(II) - ZI-ZII-ZIII-ZIV) * TAN(CONANG1)
C          XC(II) = R(II) + ED(II)
C          CG(II) = XC(II) + .5*B(II)
C          Z1=(CG(II)-XC(II))**2

```

```

      Z2=Z1/(B(II)*B(II))
      Z3=(A(II)*A(II))/(B(II)*B(II)*B(II)*B(II))
      X1=(Z1*R(II)*R(II)*Z3)/(1.-Z2)
      X2=(1.+(X1/(R(II)*R(II))))
      XG(II) = SQRT(X1/X2)
      GO TO 76
C
C      SAME CALCULATIONS FOR FOREBODY
C
79  IIII=II-1
77  B(II) = (B1*(Z(II)-ZIV))/(ZII+ZIII)
      ED(II) = ELLD1*(Z(II)-ZIV)/(ZII+ZIII)
      R(II) = (((RDIV)-XO)*(Z(II)-ZIII-ZIV))/ZII + XO
      XC(II) = (((RDIV+ELLD1)-XO1)*(Z(II)-ZIV))/(ZII+ZIII)+XO1
      A(II) = (A1*(Z(II)-ZIV))/(ZII+ZIII)
C
C      RADIUS FOUND IN REGION ZIII WHERE Z IS NOW ON THE
C      SPHERICAL NOSE
C
      IF(Z(II).LT.ZIII+ZIV) THEN
      S5=.04*D - Z(II)
      R(II)=SQRT(.0016*D*D-S5*S5)
      ENDIF
      CG(II) = XC(II) + .5*B(II)
      Z1=(CG(II)-XC(II))**2
      Z2=Z1/(B(II)*B(II))
      Z3=(A(II)*A(II))/(B(II)*B(II)*B(II)*B(II))
      X1=(Z1*R(II)*R(II)*Z3)/(1.-Z2)
      X2=(1.+(X1/(R(II)*R(II))))
      XG(II) = SQRT(X1/X2)
      II=II+1
      IF (II.GT.(NI+1-L)) GO TO 2
      IF (Z(II).LT.ZIV) GO TO 2
      IF (XC(II-1)+B(II-1).LT.R(II-1)) GO TO 2
      GO TO 77
2  CONTINUE
      III = II-1
      XX1=1.0
      WRITE(10,*) '      I      XN      J AFTER XN      CN      J AFTER CN
&YMAX'
C
C      P111=0   THE MAXIMUM OF 15 ITERATIONS IN SUBROUTINE POINTS
C              HAS NOT BEEN ACHIEVED (NEWTON-RAPHSON)
C      P111=1   THE MAXIMUM HAS BEEN ACHIEVED SO THE PROGRAM USES
C              SUBROUTINE POINTSB FROM NOW ON BECAUSE IT IS MORE
C              EFFICIENT (BISECTION).
C
      P111=0.0
      DO 3 I = 1, III
C
C      CALCULATING XN AND CN -- TANGENCY POINTS
C
      IF (P111.EQ.0.) THEN
      CALL POINTS(XC(I),A(I),B(I),R(I),XG(I),CG(I),P111,
+XN(I),CN(I))
      ENDIF

```

```

IF (P111.EQ.1.) THEN
CALL POINTSB(XC(I),A(I),B(I),R(I),XN(I),CN(I))
WRITE(*,*) '*****'
ENDIF
WRITE(*,*) I, XN(I), CN(I)

C
C      EQUATIONS TO CALCULATE THE ACTUAL POINTS WHICH WILL
C      OUTLINE THE SHIP IN THE XY PLANE IN THE FIRST QUADRANT.
C      YN IS THE Y-LOCATION OF TANGENCY POINT XN
C      DN IS THE Y-LOCATION OF TANGENCY POINT CN
C      M IS THE SLOPE OF THE LINE JOINING THE POINTS
C      B5 IS THE Y INTERCEPT OF THE LINE
C
YN(I) = SQRT(R(I)*R(I)-XN(I)*XN(I))
DN(I) = A(I)*SQRT(1.0-(((CN(I)-XC(I))/B(I))**2))
M(I) = (YN(I)-DN(I))/(XN(I)-CN(I))
B5(I) = YN(I) - M(I)*XN(I)
3 CONTINUE

C
C      X AND Y POINTS ARE CHOSEN AT CONSTANT ANGLES FOR EVERY J.
C      SUBROUTINE SPACE2 USES TWO EXPONENTIAL CURVES WHICH
C      CONNECT AND ARE C1 CONTINUOUS AT TAU. EQUATION OF CURVES IS:
C       $X = (\exp(X*K1)-1) / (\exp(K1)-1)$ 
C
K1=.5
K2=2.0
TAU=.5
ANGMAX=1.5707963
CALL SPACE2(K1,K2,TAU,NN,ANGMAX,ANG)
DO 37 J = 1, NN
ANG(J) = 1.5707963-ANG(J)
37 CONTINUE

C
C      SETTING THE X AND Y VALUES TO THEIR EXACT VALUES
C
DO 29 I = 1,III
X(I,1)=0
Y(I,1)=R(I)
X(I,NN)=XC(I)+B(I)
Y(I,NN)=0.0

C
C      BISECTION METHOD TO FIND THE VALUES OF X AND Y WHICH
C      CORRESPOND TO THE SPECIFIED ANGLES
C
DO 40 J = 2,N
XS1=0.0
XS2=XC(I)+B(I)
N5=0
30 XG1=(XS2+XS1)/2.
N5=N5+1
CALL YFIND(XG1,XN(I),CN(I),R(I),A(I),B(I),XC(I),M(I),B5(I),YG1)
TEST=ATAN(YG1/XG1)
REP=TEST-ANG(J)
IF(REP.LT.0) THEN
XS2=XG1
ELSE

```

```

        XS1=XG1
        ENDIF
        IF(N5.GT.20) GO TO 41
        GO TO 30
41 X(I,J)=XG1
   Y(I,J)=YG1
40 CONTINUE
29 CONTINUE

C
C      DEFINING THE NOSE OF THE SPACESHIP TO BE POINT (0,0,0)
C      AND DEFING THE SPHERICAL SHAPE OF THE NOSE (REGION ZIV)
C
78 DO 81 J = 1, NN
   IF (Z(II).EQ.0.0) THEN
      Z(II)=0.
      X(II,J)= 0.
      Y(II,J)= 0.
      GO TO 81
   ENDIF
   S5 = .04*D - Z(II)
   R5 = SQRT(.0016*D*D - S5*S5)
   X(II,J) = R5*COS(ANG(J))
   Y(II,J) = R5*SIN(ANG(J))
81 CONTINUE
   II=II+1
   IF (II.EQ.NI+2-L) GO TO 80
   GO TO 78
80 CONTINUE

C
C      ONLY ONE QUARTER OF THE MODEL WAS CONSTRUCTED SO THE
C      POINTS ARE EXTENDED TO CREATE HALF A MODEL.
C
13 DO 9 I = 1, NI+1-L
   DO 9 J = 1, N
      X(I,NN*2-J)=X(I,J)
      Y(I,NN*2-J)=-Y(I,J)
9 CONTINUE

C
C      PLOT3D FILE IS CREATED
C
WRITE(6,*) NI+1-L, NN*2-1, 1
DO 55 J = 1, NN*2-1
DO 55 I = 1, NI+1-L
   WRITE(6,*) Z(I)
55 CONTINUE
   WRITE(6,*) X
   WRITE(6,*) Y
   WRITE(9,*) NN*2-1, NI-L+1
DO 57 I = NI-L+1, 1, -1
DO 57 J = 1, NN*2-1
   WRITE(9,*) Z(I), X(I,J), Y(I,J)
57 CONTINUE
PPP=0.

C
C      WRITING OUT THE XNCN FILE TELLING WHERE THE LOCATIONS
C      OF THE TANGENCY POINTS ARE FOR EACH CROSS-SECTION

```

C

```

DO 204 I = III, 1, -1
DO 205 J = 1, 51
IF(PPP.EQ.2.) GO TO 205
IF(PPP.EQ.1.) GO TO 206
IF(X(I,J).GT.XN(I)) THEN
  PPP=1.
  JJ1=J
ENDIF
206 IF(X(I,J).GT.CN(I)) THEN
  PPP=2.
  JJ2=J
ENDIF
205 CONTINUE
WRITE(10,203) 202-I, XN(I), JJ1, CN(I), JJ2, XC(I)+B(I)
PPP=0.
204 CONTINUE
STOP
200 FORMAT(5X,I2,F10.3,F10.3,F10.3)
203 FORMAT(5X,I3,5X,F9.6,5X,I2,5X,F9.6,5X,I2,5X,F9.6)
END

```

C

```

SUBROUTINE POINTS(XC,A,B,R,XG,CG,P111,XN,CN)
*****
C  * SUBROUTINE TO DETERMINE THE POINTS WHERE A LINE IS TANGENT*
C  * TO BOTH A CIRCLE OF RADIUS R, AND AN ELLIPSE WITH MAJOR *
C  * AXIS B AND MINOR AXIS A USING NEWTON-RAPHSON METHOD. *
C  * XG = INITIAL GUESS FOR X ON CIRCLE *
C  * CG = INITIAL GUESS FOR X ON ELLIPSE *
C  * XN = X ON CIRCLE *
C  * CN = X ON ELLIPSE *
C  *****
P1=0.
C = CG
X = XG
P=1.
J=0
10 J=J+1
IF (X.GT.R) X=R*.999
IF (C.LT.XC) C=XC*1.001
IF(J.EQ.15) THEN
  P111=1.
  RETURN
ENDIF
Z1 = (C - XC)
Z2 = Z1/B
Z3 = ((R**2)-(X**2))
F = (Z1/SQRT(1-(Z2**2)))*(A/(B**2))-X/SQRT(Z3)
FX = ((A/(B**2))*Z1)/((1-(Z2**2))**.5)-(X**2)/(Z3**1.5)
+ -1.0/(Z3**.5)
FC = (A/(B**3))*(Z1**2)/((1-(Z2**2))**1.5)-X/(Z3**.5)+
+ (A/(B**2))/((1-(Z2**2))**.5)
G = 2.*(R**2)-2.*C*X-2*A*(((1-(Z2**2))*Z3)**.5)
GX = -2.*C+(2*X*A*(1-(Z2**2)))/(((1-(Z2**2))*Z3)**.5)
GC = -2.*X+((2.*A)/B)*Z3*Z1/(((1-(Z2**2))*Z3)**.5)
X1 = X - (F*GC-FC*G)/(FX*GC-FC*GX)

```

```

C5 = C - (F*GX-FX*G)/((FC*GX-FX*GC)*P)
IF ((C5-XC)/B.GE.1.) THEN
  P=100.
  GO TO 10
ENDIF
XG1 = ABS(X1-X)
CG1 = ABS(C5-C)
IF (XG1.GT.ABS(.005*X1)) THEN
  X=X1
  C=C5
  GO TO 10
ELSE IF (CG1.GT.ABS(.005*C5)) THEN
  X=X1
  C=C5
  GO TO 10
ENDIF
XN = X1
CN = C5
RETURN
END

C
SUBROUTINE YFIND(X,XN,CN,R,A,B,XC,M,B5,Y)
C
C *****
C * SUBROUTINE TO FIND Y GIVEN X.
C *****
REAL M
IF (X.LE.XN) THEN
  Y=SQRT(R*R-X*X)
  GO TO 25
ENDIF
IF(X.GE.CN) THEN
  Y = A*SQRT(1.0-(((X-XC)/B)**2))
  GO TO 25
ENDIF
Y = M*X + B5
25 RETURN
END

C
SUBROUTINE XFIND(Y,YN,DN,R,A,B,XC,M,B1,X)
C
C *****
C * SUBROUTINE FINDS X GIVEN Y
C *****
REAL M
IF (Y.LE.DN) THEN
  X = B*SQRT(1.0-((Y/A)**2)) + XC
  GO TO 30
ELSE IF (Y.GE.YN) THEN
  X = SQRT(R*R - Y*Y)
  GO TO 30
ELSE
  X = (Y-B1)/M
ENDIF
30 RETURN
END

C
SUBROUTINE POINTSB(XC,A,B,R,XN,CN)

```

```

C      *****
C      * SUBROUTINE TO DETERMINE THE POINTS WHERE A LINE IS TANGENT*
C      * TO BOTH A CIRCLE OF RADIUS R, AND AN ELLIPSE WITH MAJOR *
C      * AXIS B AND MINOR AXIS A BY USING THE BISECTION METHOD *
C      * C1 = INITIAL GUESS FOR X *
C      * C2 = INITIAL GUESS FOR X *
C      * XN = X ON CIRCLE *
C      * CN = X ON ELLIPSE *
C      *****
C      C1=XC
C      C2=XC+B
C      J=0
C      Z3=(A*A)/(B*B*B*B)
10  J=J+1
C      C = (C1+C2)/2.
C      Z1=(C-XC)**2
C      Z2=Z1/(B*B)
C      X1=(Z1*R*R*Z3)/(1.-Z2)
C      X2=(1.+(X1/(R*R)))
C      X=SQRT(X1/X2)
C      G = R*R - C*X - A*SQRT((1.-Z2)*(R*R-X*X))
C      G = ((C-XC)*A)/((SQRT(1-Z3))*B*B)-(X/(SQRT(Z1)))
C      IF (G.GT.0.0) THEN
C          C1=C
C      ELSE
C          C2=C
C      ENDIF
C      IF (J.LT.20) GO TO 10
C      XN = X
C      CN = C
C      RETURN
C      END

SUBROUTINE SPACE2(K1,K2,TAU,N,ANGMAX,ANG)
C      *****
C      * SUBROUTINE USES TWO EXPONENTIAL CURVES TO GIVE A *
C      * DISTRIBUTION FOR THE ANGLES USED IN PROGRAM *
C      *****
C      DIMENSION ANG(N)
C      REAL K1,K2,K3,NOM
C      ICOUNT=0
C      K3=-K2
300  CONTINUE
C      TEMP=K1*K2*(1-TAU)/((1-K1)*TAU)
C      DETAL=1./(1-EXP(-K2))
C      DETAR=K3/(EXP(K3)-1.)
C      C=TEMP*DETAL
C      DF=DETAR-C
C      DFDK=(EXP(K3)-1.-K3*EXP(K3))/(EXP(K3)-1)**2
C      DK3=-DF/DFDK
C      IF(ABS(DK3).LT..00001) GO TO 200
C      K3=K3+DK3
C      ICOUNT=ICOUNT+1
C      IF(ICOUNT.GT.20) GO TO 200
C      GO TO 300
200  CONTINUE

```

```
SCALEX=ANGMAX
SCALEF=SCALEX
1  CONTINUE
   DO 10 I=1,N
   ETA=FLOAT(I-1)/FLOAT(N-1)
   IF(ETA.GE.TAU) GO TO 2
   TERM=K2/TAU
   NOM=EXP(TERM*ETA)-1.
   DNOM=EXP(K2)-1.
   F=K1*(NOM/DNOM)
   GO TO 3
2  CONTINUE
   TERM=K3/(1-TAU)
   NOM=EXP(TERM*ETA-TERM*TAU) -1.
   DNOM=EXP(K3)-1
   F=K1+(1.-K1)*NOM/DNOM
3  CONTINUE
   IX=SCALEX*ETA
   ANG(I)=SCALEF*F
10 CONTINUE
11 CONTINUE
   RETURN
   END
```

